Solution Of The Variable Coefficient Poisson Equation On Cartesian Hierarchical Meshes In Parallel: Applications To Phase Changing Materials

Alice Raeli

THÈSE PRÉSENTÉE POUR OBTENIR LE GRADE DE

DOCTEUR DE L'UNIVERSITÉ DE BORDEAUX

ÈCOLE DOCTORALE DE MATHÉMATIQUES ET INFORMATIQUE
SPÉCIALITÉ MATHÉMATIQUES APPLIQUÉES
Memphis Team Projects

Par Alice RAELI

Solution Of The Variable Coefficient Poisson Equation On Cartesian Hierarchical Meshes In Parallel: Applications To Phase Changing Materials

---

Sous la direction de:
Angelo IOLLO, Professor, University of Bordeaux
Mejdi AZAÏEZ, Professor, University of Bordeaux
Michel BERGMANN, Research Scientist (CR), University of Bordeaux

Membres du jury:
Mr. BRUNEAU Charles-Henri, Professor, University of Bordeaux, President of the Jury
Mr. DI PIETRO Daniele Antonio, Professor, University of Montpellier, Referee
Mr. GOLAY Frédéric, Maître de Conference, University of Toulon, Referee
Mr. LANTERI Stéphane, Research Scientist, INRIA Sophia Antipolis, Examiner
Mr. RUSSO Giovanni, Professor, University of Catania, Examiner

5^{th} October 2017

"*Il miglior riconoscimento per la fatica fatta non è ciò che se ne ricava, ma ciò che si diventa grazie a essa.*"
John Ruskin
"*E questo mi riporta a Dennis Ritchie. La nostra collaborazione è stata una questione di bellezza.*"
Ken Thompson

Solution du problème de Poisson avec coefficients variables sur maillages cartésiens hiérarchiques en parallèle: applications aux matériaux avec changement de phase

Résumé Détaillé

Les matériaux avec changement de phase (PCM) sont utilisés pour le stockage de l'énergie solaire grâce à leur forte capacité thermique. Ces matériaux sont également caractérisés par leur faible conductivité thermique. Les industriels utilisent des mélanges de PCM (graphite et sel comme mentionné par la suite, ou autres...) avec des milieux caractérisés par une haute conductivité thermique. Le matériau hybride qui en résulte (Fig. 1) présente ainsi à la fois une haute conductivité et une forte capacité thermique.

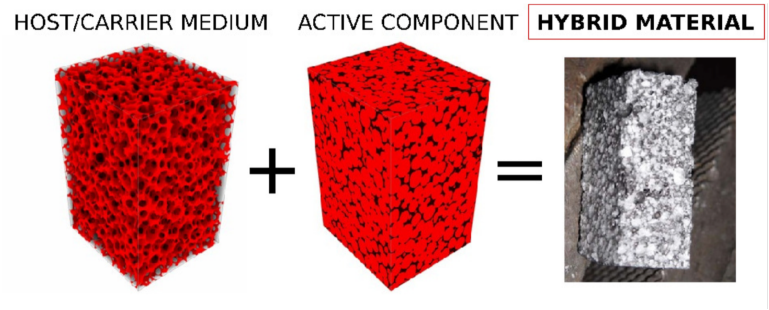


Figure 1: Matériau hybride

Le matériau testé lors de cette thése est un mélange de graphite et de PCM, il est utilisé par l'entreprise ABEGNOA Solar (Fig.2). Cette société travaille sur les énergies renouvelables et l'environnement durable.

Figure 2: PS10 Solar Power Plant, ABENGOA Solar - Seville, Espagne

La structure du graphite considérée ici contient des capsules salines de matériau qui passe de la phase solide à la phase liquide. Le processus de stockage d'énergie s'effectue par changements de phase du PCM. Si le PCM

se liquéfie en période d'ensoleillement et capte la chaleur, il se solidifie ensuite la nuit et restitue cette chaleur sous forme d'énergie thermique. Ce modèle présente une première difficulté lors de la conception géométrique, en particulier pour le positionnement des capsules. Si elles sont trop proches ou trop petites, le matériau risque de s'évaporer (dû à la porosité du graphite). A l'inverse, si elles sont trop grosses ou trop lointaines, il ne se liquéfiera pas totalement et la quantité d'énergie stockée ne sera pas maximale.

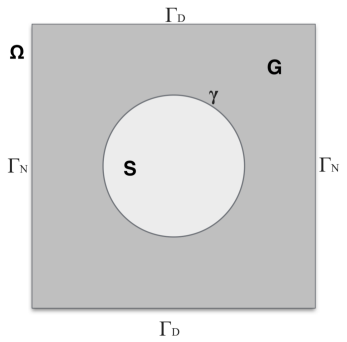


Figure 3: Configuration simplifiée d'étude.

On s'intéresse en particulier à l'analyse des discontinuités de contact entre les deux milieux (entre le graphite et le PCM). Dans cette thèse, nous présentons une approche qui se concentre sur ces zones, où le milieu hybride est dans un domaine Ω, avec G et S les parties qui représentent respectivement le graphite et le sel PCM (Fig. 3). On obtient $\Omega = G \cup S$ et on appelle γ la surface entre les deux matériaux, en faisant la distinction entre les conditions aux limites de Neumann Γ_N et de Dirichlet Γ_D. En notant u la température, les équations qui modélisent l'état stationnaire de notre problème physique sont:

$$
\begin{aligned}
-\mathrm{div}(k(\boldsymbol{x}) \bigtriangledown u(\boldsymbol{x})) &= g(\boldsymbol{x}), && \text{in } G \cup S, && \text{(1a)}\\
k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, && \text{on } \Gamma_N, && \text{(1b)}\\
u(\boldsymbol{x}) &= u_D(\boldsymbol{x}), && \text{on } \Gamma_D, && \text{(1c)}\\
[k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})] &= 0, \quad R(k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S = [u(\boldsymbol{x})], && \text{on } \gamma, && \text{(1d)}
\end{aligned}
$$

où le vecteur normal unitaire $\boldsymbol{n}$ pointe vers l'extérieur du milieu S et $[\cdot]$ représente le saut à travers l'interface γ. La conductivité thermique $k(\boldsymbol{x}) = (k_G, k_S(u))$ peut prendre différentes valeurs entre S et G. Nous considérons dans le cadre de ce travail un changement brutal entre les deux matériaux. Le premier chapitre de cette thèse sera donc dédié à l'étude du problème de Poisson avec coefficients variables et discontinuités sur des sous-domaines. On présentera les méthodes développées et utilisées pour la résolution numérique.

Lors de la résolution des équations aux dérivées partielles (EDP), une tâche fondamentale consiste à déterminer la structure discrète utilisée pour représenter le domaine continu. Dans cette thèse, nous présenterons une méthode compacte aux différences finies sur des maillages hiérarchiques cartésiens structurés quadtrees/octrees en deux et trois dimensions. Les méthodes de raffinement adaptatif (AMR) ont pour objectif commun de limiter les coûts en mémoire et en temps de calcul, ainsi que d'obtenir une précision accrue dans les zones d'intérêt du domaine (les zones de discontinuités ou de sauts). Dans le deuxième chapitre de cette thèse, nous présenterons les structures octree afin d'en montrer les avantages en terme de calcul ainsi que les difficultés observées. Notre code est prédisposé pour être développé en parallèle grâce à la structure de données utilisant le *Z-order* pour la numérotation des élements du maillage.

Par la suite, nous considérerons une méthode aux différences finies où les inconnues sont au centre des cellules. Nous utilisons tous les voisins d'un octant pour calculer les poids du schéma afin d'en garantir la consistance et d'assurer une convergence à l'ordre deux dans la mesure du possible. Ce processus nécessite la résolution d'un problème d'optimisation dans lequel nous minimisons l'erreur de troncature sur chaque octant en fonction de son voisinage.

Les résultats principaux concernant la méthode sont présentés dans le quatrième chapitre en 2D et 3D. La première partie est dédiée à l'étude de convergence du schéma, en particulier en 2D. On considère une configuration de référence qui ne converge pas si l'on prend uniquement une partie des voisins possibles, et nous étudions la convergence après application du schema. Cette configuration a été également comparée à deux méthodes aux volumes finis, développées par des membres de notre équipe. En 3D, nous proposons les études de convergence du Laplacien et des résidus. La discrétisation du Laplacien utilisant le Z-order génère une structure de matrice dispersée, difficile à appréhender en l'absence d'un solveur particulièrement adapté. Cependant, nous montrerons sur des exemples multi-échelles que l'AMR octree mis en place garantit des gains de temps et de mémoire CPU significatifs par rapport à des maillages cartésiens uniformes. Les tests montrent aussi que globalement, le schéma tend vers l'ordre deux en l'absence de discontinuités (pénalisation). On présente aussi un premier cas de discontinuité du terme de conductivité thermique traité comme une fonction continue qui varie brusquement entre deux valeurs. Le chapitre se conclut avec une présentation plus détaillée du code de calcul en insistant sur les propriétés de parallélisation unitlisées, sur les librairies concernées pour le calcul, et enfin sur une étude de scalabilité forte et faible pour démontrer les propriétés en temps de la résolution. Le dernier chapitre de cette thèse contient une introduction à l'équation de la chaleur, en particulier pour la diffusion de chaleur dans les matériaux hybrides mentionnés ci-dessus. On introduit un terme instationnaire qui permet de décrire l'évolution de l'état

d'un système sujet au changement de phase pour chaque pas de temps. Ce terme d'enthalpie s'ajoute au problème (1), ce qui conduit au système d'équations suivant:

$$\begin{aligned}
\partial_t H(u(\boldsymbol{x})) - \nabla \cdot (\kappa(\boldsymbol{x}) \bigtriangledown u(\boldsymbol{x})) &= 0, && \text{in } G \cup S,\\
\kappa(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, && \text{on } \Gamma_N,\\
u(\boldsymbol{x}) &= u_D, && \text{on } \Gamma_D,\\
[k(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x})] &= 0, && \text{on } \gamma,\\
R(k(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S &= [u(\boldsymbol{x})], && \text{on } \gamma.
\end{aligned}$$

Pour résoudre ce problème de façon implicite en temps nous utilisons une projection avec un pseudo pas de temps (méthode de relaxation). Cette approche nous permettra d'étudier le temps de fusion des capsules PCM en fonction de la surface occupée en pourcentage dans son conteneur conducteur (Fig. 4).

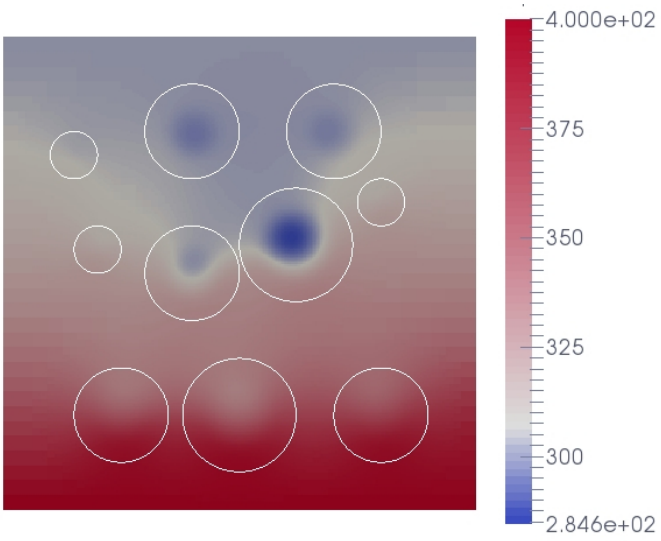


Figure 4: Exemple d'évolution de la chaleur au cours du temps. Cas avec dix capsules de PCM.

Mots clés: AMR, octree, différences fines, discrétisation équations aux dérivées partielles, équation de la chaleur, discontinuités intérieures.

Unité de recherche
Institut de Mathématiques de Bordeaux, UMR 5251,
351 cours de la Libération, 33405 TALENCE cedex.

Acknowledgements

I would like to express my gratitude to my thesis directors for their support. Professor Iollo Angelo, Professor Azaïez Mejdi and Chief Researcher Bergmann Michel, each of them in these years has been the pillars of my professional and personal growth. I have had the opportunity to devote myself freely to what is more exciting in research, and at the same time they have helped me in all those moments of a doctorate that may seem obscure.

I would like to thank Professors Di Pietro Daniele Antonio and Golay Fréderic for having kindly accepted to be referees of this thesis. I am also grateful to Professors Bruneau Charles-Henri and Lanteri Stéphane for agreeing to be members of the jury; I am particularly grateful to Professor Russo Giovanni, without whom I probably would not have had this experience, for making my interest in scientific research vivid with me with his innovative and professional spirit.

To my parents, Angela and Stefano, for their support and their trust in me. They always encouraged me and showed constant love.

To my boyfriend Sebastiano, for supporting me, even just listening to my French presentations over and over again for hours without knowing either maths or French. Also for every time he encouraged me, often surprising me more than I could imagine, as a person and as an unexpected maigret de canard cook.

To my colleagues: with them I have had the pleasure of working in a serene and sunny environment, I had valuable and constructive personal and working relationships with each of them. To the patience and to the ever-useful suggestions that they have been able to give me; to the anxieties shared and fuelled by many of them, including Emanuela and Federico in particular: with them it was easy to increase immature fears. To my *old* and also to those *new* friends met in Bordeaux, they all will stay in my heart also after this experience. To all the wonderful people I've met in a list of names that might become too long.

Thank you all.

This thesis was funded by the University of Bordeaux.

This work is supported by a public grant overseen by the French National Research Agency (ANR) as part of the progam "*Investissements d'Avenir*" (reference: ANR-10-LABX-0083 Labex EFL)

Ringraziamenti

Vorrei esprimere la mia gratitudine ai miei direttori di tesi per il loro supporto. Il Professore Angelo Iollo, il Professore Mejdi Azaïez ed il Ricercatore Michel Bergmann. Ognuno di loro in questi anni è stato pilastro portante della mia crescita professionale e personale. Ho avuto l'opportunità di dedicarmi liberamente a quel che più mi appassiona nella ricerca ed allo stesso tempo mi sono sentita seguita nei miei dubbi e nelle mie imperfezioni, persino soccorsa in quei momenti di un dottorato che possono sembrare quantomai oscuri prima che una piega proficua si profili all'orizzonte.

Vorrei ringraziare i Professori Di Pietro Daniele Antonio e Golay Fréderic per aver gentilmente accettato di essere referenti nel valutare la mia tesi. Ringrazio per aver acconsentito ad esser membri della giuria i Professori Bruneau Charles-Henri e Lanteri Stéphane; particolare menzione dedico al Professore Russo Giovanni, senza il quale probabilmente non avrei vissuto questa esperienza, per aver reso vivido in me l' interesse per la ricerca scientifica con il suo animo innovativo e professionale.

Ai miei genitori, Angela e Stefano, per il loro sostegno, per la fiducia in me con cui mi hanno sempre incoraggiata e per l'amore costante, che mi hanno mostrato con ogni piccolezza che non hanno mai smesso di dare per scontata nel loro esprimere affetto. Li ringrazio per aver sopportato con me alti e bassi di questa distanza, moralmente e fisicamente difficile.

Al mio ragazzo Sebastiano, per avermi supportato in ogni modo, anche solo ascoltando più e più volte per ore le mie presentazioni in francese senza sapere nè la matematica nè il francese, solo per dimostrarmi affetto in questa distanza, sostegno e pazienza. Per ogni bel momento vissuto in cui mi ha incoraggiata riuscendo spesso a sorprendermi più di quanto potessi immaginare, come persona e come inatteso cuoco di maigret de canard.

Ai miei colleghi a cui devo il piacere di aver lavorato in un ambiente sereno e solare, per ognuno di loro vi sono stati confronti personali e lavorativi pregevoli e costruttivi. Alla pazienza ed ai consigli sempre utili che sono stati capaci di darmi, prestandomi spesso disponibilità ad uno sfogo o ad un'incertezza; alle ansie condivise e alimentate con molti di loro, tra cui Emanuela e Federico in particolare con i quali accrescere vicendevolmente immotivati timori è stato semplice. Ai miei amici *vecchi* ed anche a quelli *nuovi*, frutto di questa esperienza a Bordeaux ma capaci di restare nel cuore per ben oltre la stessa. A tutte le splendide persone che ho conosciuto in una lista di nomi che potrebbe divenire troppo lunga.

A chi, per me importante, purtroppo non c'è più.

Grazie a tutti.

Contents

Introduction

Context: The Phase Changing Materials

Phase-changing materials (PCM) are used in various industrial applications for storing energy due to their high thermal capacity. On the other hand these materials usually have very low thermal conductivity. Several techniques involving mixing the PCM with other highly conductive materials allow this drawback to be overcome. The resulting composite material has both high thermal conductivity and high thermal capacity. Depending on the technique, the shape of the interface between the different components of the composite material is more or less complex.

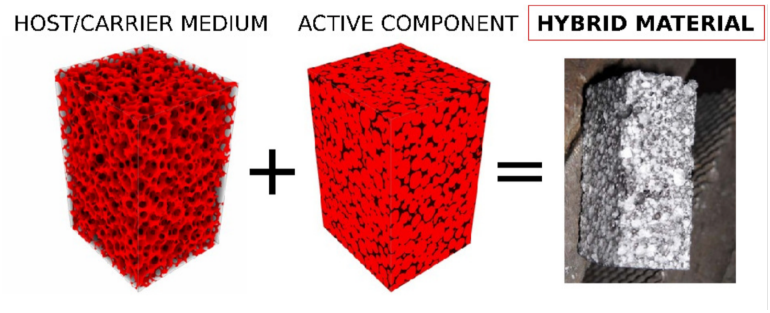


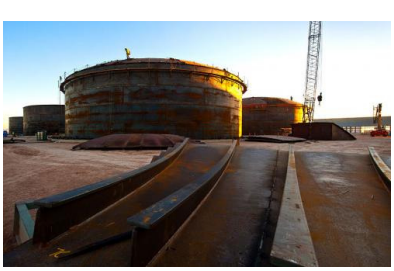
Figure 5: Composite material

The model that we studied in this thesis is related to a graphite/salt composite material (Fig.5) belonging to ABENGOA Solar (Fig.6); this company is involved in renewable and sustainable energy.

The material to which we are referring has a graphite structure containing salt capsules, which are subjected to a phase change from solid to liquid. The ideal loop of this energy storage system requires that the salt liquefies during the day, preserving its temperature, then solidifies during the night and it releases heat to the graphite, converting it in energy.

The produced material is conserved in tanks (Fig. 7a) where heat is transported during the day from the heliostats area using nanotubes that direct hot gases. These nanotubes cross sections of the hybrid material by dissipating the heat to the graphite they are in contact with (Fig. 7b). This

Figure 6: The PS10 Solar Power Plant, ABENGOA Solar - Sevilla, Spain



(a)

(b)

Figure 7: Tank (left) containing the PCM and a section of the material (right).

model presents the first difficulties in the provisions of salt capsules; if they are too small, the PCM evaporates and consequently is dispersed; moreover, if they are too big or close together, the PCM does not reach the liquid state entirely.

In the study of these materials it is particularly important to analyse the surface contact between the different components in the internal interfaces. In this thesis we show how to manage localized refinements on the zones concerned by this kind of variations; our approach will be initially more general, then gradually we add new properties to our problem until the model describes the material at best.

Heat Conduction Problem

We describe a hybrid material on a simplified domain Ω. Let G be the host material of the presented mixture (graphite, for example) and S be the one that changes its phase (PCM) (Fig. 8). We have $\Omega = G \cup S$, and we denote with γ the interface between both sub-domains. We distinguish the tank boundaries into an isolated part (Neumann boundary conditions Γ_N) and a conductive one (Dirichlet boundary conditions Γ_D). Assuming that u stands for temperature, we impose u_0 as initial condition when the entire system

is turned off.

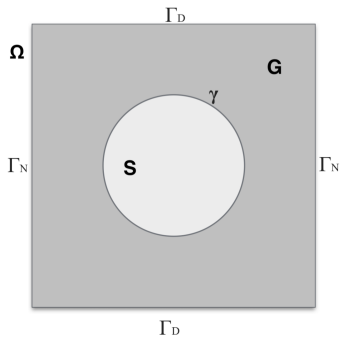


Figure 8: Simplified domain example.

The equations describing the steady-state conduction in Ω are:

$$
\begin{aligned}
-\mathrm{div}(k(\boldsymbol{x}) \bigtriangledown u(\boldsymbol{x})) &= g(\boldsymbol{x}), && \text{in } G \cup S, && \text{(2a)}\\
k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, && \text{on } \Gamma_N, && \text{(2b)}\\
u(\boldsymbol{x}) &= u_D(\boldsymbol{x}), && \text{on } \Gamma_D, && \text{(2c)}\\
[k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})] &= 0, \quad R(k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S = [u(\boldsymbol{x})], && \text{on } \gamma, && \text{(2d)}
\end{aligned}
$$

where $\boldsymbol{n}$ is the normal vector oriented outward from S, and the symbol $[\cdot]$ refers to the jump through γ. We call R the contact resistance between the two components. The thermal conductivity represented by $k(\boldsymbol{x})$ can be different in G and S. In our cases it changes significantly ($k(\boldsymbol{x}) = (k_G, k_S(u))$).

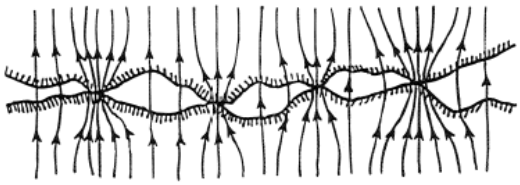


Figure 9: Empty layer generated by the loss of volume in liquid state.

In Figure 9 a graphical explanation of the conditions on γ (eq. (2d)) is proposed: the loss of volume caused by the liquid state generates an empty layer, across which the temperature undergoes a discontinuity; however the flux conservation condition ensures the continuity of the normal derivatives. In this work, we take into account the steady state in the first part; otherwise, in the future the eq. (2a) has an added term, $\partial_t H(u)$, which depends on the two materials and the phase change conditions and it controls the latent heat. This function will be presented in detail in Chapter 5. Moreover,

in the first part of this thesis, we suppose that the conductivity term $k(\boldsymbol{x})$ is constant, so we will reduce it to the Poisson equation $k(\boldsymbol{x})\Delta u(\boldsymbol{x}) = g(\boldsymbol{x})$ on the entire domain. A survey of discretizations for these kinds of problems is given in Chapter 1.

Numerical Method

In the resolution of partial differential equations (PDEs) problems, discrete domains are used that describe the continuous one on which the solution lies. One method is to approximate the unknown values on grid points that cover the domain in a specific way. The position of grid points determines the local error and, hence, the accuracy of the solution. The spacing between adjacent points also determines the cost of the computation.

The correct choice of the data structure can decrease the time complexity of a resolution. In the general case, a uniform mesh is efficient, but there are special cases where the solution is more difficult to estimate in some regions (perhaps due to discontinuities, steep gradients, etc.) than in others. A uniform grid with a finer refinement to catch these difficult regions can be acceptable, but this approach is computationally costly. Many problems in numerical analysis, however, do not require a uniform precision in the entire discretized domain; from this observation, it is necessary to focus the computation on specific areas of interest that require precision.

In numerical analysis, adaptive mesh refinement, or AMR, is a method of adapting the accuracy of a solution within certain sensitive regions of the entire domain, dynamically during the calculation or fixed statically at its beginning. It provides focus on the precision of the numerical computation based on those areas while leaving the other regions of the domain at lower levels of precision and resolution. This technique has been accredited to Marsha Berger, Joseph Oliger, and Phillip Colella who developed an algorithm for dynamic grids called local adaptive mesh refinement, which begins with the entire computational domain being covered with a coarsely resolved base-level regular Cartesian grid. As the calculation progresses, individual grid cells are tagged for refinement, and the computation follows the internal jumps generated over the domain.

The AMR approach allows the user to solve problems that are completely intractable on a uniform grid. In this thesis, a numerical algorithm is presented for solving elliptic partial differential equations subjected to discontinuities. The computational domain is subdivided into Cartesian, hierarchical, quadtree-based meshes, above which a cell-centred discretization is devised.

Several techniques to numerically solve partial differential equations exist. The finite element and finite volume methods are widely used in engineering and in computational fluid dynamics, and they are easily adapted

to problems with complicated geometries. Other methods are often more accurate, but they provide smooth solutions. The method chosen for the current work is a finite difference method; in general, this method is simpler to use and understand. Our approach was conceived to be optimal for adaptive mesh refinement approaches, and the respective computation was created to be easily managed in parallel.

Structure of the thesis

The **Chapter 1** it is a survey of Poisson equations on complex geometries and also with variable coefficients. The chapter contains the analytical basis of the problem, with existence and uniqueness results for solutions. We recall a variational formulation for the above specific problem (2), which will be studied in the following chapters. Various discretizations are considered in order to introduce the problem within which the method presented in this thesis is placed.

In **Chapter 2** we present the octree data structure, on which we devise our numerical method. The aim of this chapter is to outline the purposes and advantages of these kinds of hierarchical meshes, explaining technical details that recur many applications. We introduce different computing categories where octrees are used, involving AMR. Then we make an accurate study of mesh ordering to complete the general framework of our structure usage and improve understanding of the algorithms used in computation. This part ends with the first employment of our application to uniquely identify the configurations that can occur. This tool in particular, when used, is necessary to gain computing time and avoid some interprocess communications.

In **Chapter 3** the finite difference methods technique is presented in general; then we construct our method in detail. The algorithm is devised to impose the consistency constraints and to minimize the second-order accuracy by studying each configuration independently from others. We explain the minimization problem's creation, concluding with a comparison with uniform mesh case. In general, it is difficult to prove the stability of finite difference methods, such as the one presented, so we justify the choice of using a five-points weights distribution when the uniform configuration occurs, in front of a second-order accurate nine-points stencil computed with our method. Other finite difference methods are also presented. We highlight that our method allows us to examine each configuration on its intrinsic geometry; meanwhile, most existing methods tend to interpolate ghost neighbour values to recreate a fictitious semi-classical configuration.

In **Chapter 4** we present the main results of our method in two and three dimensions. The first part is devoted to the consistency study, as a proof of the numerical convergence of the method globally, taking, for example, spe-

cific two-dimensional cases. There is also a comparison of the method with two finite volume methods implemented within our work group. We show cases of unbalanced meshes to reinforce the study of weights computed with our strategy. In three dimensions, we then show a convergence study of the residual of the operator, analysing its norm; it then follows the numerical consistency tests on Laplacian equation. A progressive study is proposed to reach the discontinuous model, starting from penalization and ending with a study on discontinuities added in the Poisson equation. In this part, the comparison between our AMR method and a uniform resolution is outlined, highlighting the advantages in time and memory at the same error order. The chapter ends with a code presentation: its internal properties, the libraries involved, and the optimized parts; the scalability tests are also reported to confirm the properties of the parallelism used.

In **Chapter 5** the application model about phase changing material simulations is presented, including a study of modellisation and an outline of the difficulties that it involves. A first jump result is shown through a fictitious layer model that simulates resistance due to thermal conductivity discontinuity; cases of the model involving time evolution are proposed, concluding the robustness proof of our method from the Laplace functional to the heat equation with internal discontinuities.

Chapter 1

Poisson Problems On Complex Geometries

We introduced the problem model, whose more complex parts are given by strong internal discontinuities and by having to solve the equation with variable coefficients on complex geometries. In this chapter, we present this problem and its difficulties.

Partial differential equations (PDEs) often describe physical phenomena existing in nature, or man-made; however this ability to model real-life applications can lead to complex geometric structure to delineate. This thesis represents an example of heat conduction problem with mixed materials interaction, but the vast set of applications includes tumours in human bodies, electrodynamics problems, fluid dynamics, quantum mechanics and wave theory.

To face complex geometrical problems, standard discretization methods require multilevel solutions that can result in being hard to compute; moreover the geometry might evolve in time, which would require a new discretization at each time step. Thus, most methods to approach these problems are non-standard and based on the complex geometry given explicitly or implicitly by level set functions. Other methods preserve the numerical scheme and are simple to manage, acting on the domain directly as diffused domain methods.

In this part, we present an analytical characterization of the equations that we refer to, and different discretizations used to resolve them, highlighting their different computational properties.

1.1 The Poisson Problem

A wide set of physical problems are represented by Poisson's equations; its origins are generally associated with the study in the fields of mechanics, physics of conducting media, electrostatics and gravitational potential. The

applicability of the Poisson problem also includes other disciplines. Several different methods have been studied to solve this kind of equation of the form:

$$\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}) \quad \text{on } \Omega \subset \mathbb{R}^d, \tag{1.1}$$

with $d \leq 1$, $f : \Omega \to \mathbb{R}$.

The homogeneous equivalent of the Poisson equation is the *Laplace equation*:

$$\Delta u(\boldsymbol{x}) = 0.$$

Definition 1.1.1 *A function $u(\boldsymbol{x}) : \Omega \to \mathbb{R}$ is called* harmonic *when it satisfies the Laplace equation $\forall \boldsymbol{x} \in \Omega$.*

Harmonic functions have special smooth geometrical properties, in general; for problem derivation and solution properties, we invite the interested reader to read [1]-[2].

For $\Omega \subset \mathbb{R}^2$, rigid motion or isometric deformation of Ω don't change the value of the Laplacian operator Δ. In the following, we will denote with u the term $u(\boldsymbol{x})$ for simplicity.

1.1.1 The Boundary Conditions

We can consider a Poisson problem with different possible boundary conditions:

$$\alpha u + \beta(\nabla u \cdot \boldsymbol{n}) = g \quad \text{on } \partial\Omega, \tag{1.2}$$

where α and β are constant, although variable coefficients are possible. $\boldsymbol{n}$ is the outward unit normal vector and $g : \partial\Omega \to \mathbb{R}$.

The corresponding problem (1.1)-(1.2) is considered **ill-posed** if the boundary conditions imposed are either not sufficient to yield a unique solution to the problem or over-constrained so that no non-trivial solution to the problem exists.

Dirichlet Boundary Condition

When β is zero, the boundary condition is of *Dirichlet* type

$$u = g_D \quad \text{on } \partial\Omega. \tag{1.3}$$

The physical interpretation of the Laplace equation with Dirichlet boundary conditions (1.1)-(1.3) is a heat application through the surface of a domain which will reach a temperature inside up to a steady state.

Neumann Boundary Condition

If α, in (1.2) is equal to zero, the condition is of *Neumann* type

$$\nabla u \cdot \boldsymbol{n} = g_N \quad \text{on } \partial\Omega. \tag{1.4}$$

When $g_N = 0$, we talk about *natural* boundary condition. It represents on the steady state, for equivalent heat problem representation, that the boundary is totally insulated from external sources.

Cauchy and Robin Boundary Condition

When α and β in (1.2) are both nonzero

$$\alpha u = g_1 \quad \text{on } \partial\Omega,$$

$$\beta(\nabla u \cdot \boldsymbol{n}) = g_2 \quad \text{on } \partial\Omega,$$

the boundary condition is of *Cauchy* type.

In general, this condition generates an ill-posed problem and does not guarantee a solution for elliptic problems; however each class of PDE requires a different class of boundary conditions in order to have a unique, stable solution. Even if we are not dealing with it in this work, for example hyperbolic equations require the Cauchy boundary condition for existence and uniqueness; meanwhile, others are not sufficient or are too restrictive (see also [3]). The Cauchy boundary condition is analogous to the initial conditions for a second-order ordinary differential equation, and it corresponds to imposing both a Dirichlet and a Neumann boundary condition.

Less common, but provided in many application, is the *Robin* boundary condition

$$\alpha u + \beta(\nabla u \cdot \boldsymbol{n}) = g \quad \text{on } \partial\Omega,\ \alpha \neq 0, \beta \neq 0,$$

where the value of a linear combination of the dependent variable and the normal derivative of the dependent variable is specified on the boundary.

Mixed Boundary Conditions

A third possibility is that Dirichlet conditions are valid on part of the boundary $\partial\Omega_D$, and Neumann conditions hold on the relative complement of Ω with respect to Ω_D, *i.e.* $\partial\Omega_N = \partial\Omega / \partial\Omega_D$:

$$u = g_D \quad \text{on } \partial\Omega_D, \tag{1.5}$$

$$\nabla u \cdot \boldsymbol{n} = g_N \quad \text{on } \partial\Omega_N. \tag{1.6}$$

Most numerical tests presented in this thesis are boundary value problems with Dirichlet or mixed boundary conditions, thus it is necessary to list some results about the solutions of these problems.

1.1.2 Uniqueness of Solutions to the Poisson Equation

For the case of Dirichlet boundary conditions or mixed boundary conditions, the solution to Poisson's equation always exists and is unique when Ω is compact, smooth, and with a piecewise smooth boundary. Finally, for the case of the Neumann boundary condition, a solution may or may not exist (depending on whether a certain condition is satisfied). If a solution exists, then it is unique up to an overall additive constant.

For the existence of mentioned solutions, we invite the interested reader to read [4], where a Green's function complete analysis offers solution existence proofs using the *magic rule*. The author also proposes an accurate explanation for the instability of Cauchy boundary conditions, with respect to the fact that solution existence could not be proved in general. More extensive results for elliptic operators, solution existence, local properties and continuities are given by S. Agmon in [5]. The author proves local existence and regularity, then introduces Garding's inequality to ensure the global existence for these operators.

The Uniqueness For Dirichlet Boundary Conditions

Consider the problem (1.1)-(1.3). If u_1 and u_2 are both solutions, with $u_1 \neq u_2$, they satisfy the same boundary condition. Then $u_1 = u_2$ on $\partial\Omega$.

Consider the function $\phi = u_1 - u_2$. It satisfies:

$$\Delta\phi = 0 \text{ on } \Omega, \tag{1.7}$$

$$\phi = 0 \text{ on } \partial\Omega. \tag{1.8}$$

We assume that $\phi \neq c, \forall c \in \mathbb{R}/0$ because if ϕ is constant, the boundary condition (1.8) imposes has to be the zero-function over the entire domain, and it is impossible by assumption. As a consequence the following integral has a positive contribution

$$I = \int_\Omega (\nabla\phi)^2 \, d\Omega > 0. \tag{1.9}$$

Moreover, we can rewrite by the product rule

$$\nabla \cdot (\phi\nabla\phi) = (\nabla\phi)^2 + \phi\Delta\phi \quad \xrightarrow{(1.7)} \quad \nabla \cdot (\phi\nabla\phi) = (\nabla\phi)^2. \tag{1.10}$$

Using this result in the above integral formula and the divergence theorem to convert the integral over the closed surface $\partial\Omega$, we obtain:

$$I = \int_\Omega \nabla \cdot (\phi\nabla\phi) \, d\Omega = \int_{\partial\Omega} \phi(\boldsymbol{n} \cdot \nabla\phi) \, d(\partial\Omega). \tag{1.11}$$

We remark (1.8); hence, it follows that $I = 0$ but is possible only if ϕ is a constant inside Ω; consequently, this constant must be zero since $\phi = 0$ on $\partial\Omega$, which is equivalent to the statement that $u_1 = u_2,$ on Ω.

It follows that, if the Poisson problem with Dirichlet boundary condition has a solution, it has to be the unique one.

The Uniqueness For Neumann Boundary Conditions

Consider the problem (1.1)-(1.4). If u_1 and u_2 are both solutions, with $u_1 \neq u_2$, they satisfy the same boundary condition. Then $\boldsymbol{n} \cdot \nabla u_1 = \boldsymbol{n} \cdot \nabla u_2$ on $\partial\Omega$.

Consider the function $\phi = u_1 - u_2$. It satisfies:

$$\Delta\phi = 0 \text{ on } \Omega, \tag{1.12}$$

$$\boldsymbol{n} \cdot \nabla\phi = 0 \text{ on } \partial\Omega. \tag{1.13}$$

As done for the Dirichlet boundary condition case, we consider the integral:

$$I = \int_\Omega (\nabla\phi)^2 \, d\Omega. \tag{1.14}$$

Moreover, a simple product rule:

$$\nabla \cdot (\phi\nabla\phi) = (\nabla\phi)^2 + \phi\Delta\phi \quad \xRightarrow{(1.12)} \quad \nabla \cdot (\phi\nabla\phi) = (\nabla\phi)^2 \tag{1.15}$$

Using this result in the integral formula and the divergence theorem to convert the integral over the closed surface $\partial\Omega$, we obtain:

$$I = \int_\Omega \nabla \cdot (\phi\nabla\phi) \, d\Omega = \int_{\partial\Omega} \phi(\boldsymbol{n} \cdot \nabla\phi) \, d(\partial\Omega). \tag{1.16}$$

We remark that (1.13); hence, it follows that $I = 0$, but it is possible only if ϕ is a constant inside Ω. Consequently, $\phi = c \in \mathbb{R},$ on Ω, which is equivalent to the statement that $u_1 - u_2 = c,$ on Ω.

It follows that if the Poisson problem with Neumann boundary condition has solutions, they at most differ for a constant value. However, there is an important consistency condition that must be satisfied in order for a solution to the Neumann boundary value problem to exist.

We integrate equation (1.1), and we apply the divergence theorem as follows:

$$\int_\Omega \Delta u \, d\Omega = \int_\Omega f \, d\Omega \tag{1.17}$$

$$\Downarrow$$

$$\int_\Omega f \, d\Omega = \int_\Omega \nabla \cdot \nabla u \, d\Omega = \int_{\partial\Omega} (\boldsymbol{n} \cdot \nabla u) \, d(\partial\Omega). \tag{1.18}$$

The term $\int_\Omega f \, d\Omega$ is known a priori; we can then conclude that solutions exist and differ for a constant value, if and only if the Neumann boundary condition satisfies (1.18).

The Uniqueness For Mixed Boundary Conditions

In the case of mixed boundary conditions

$$u = g_D \quad \text{on } \partial\Omega_D, \tag{1.19}$$
$$\nabla u \cdot \boldsymbol{n} = g_N \quad \text{on } \partial\Omega_N, \tag{1.20}$$

we can again use the above procedure using $\phi = u_1 - u_2$, and the integral $I = \int_\Omega (\nabla\phi)^2 \, d\Omega$, to conclude that if u_1 and u_2 are both solutions, then ϕ is a constant on Ω. Being zero on a part of the boundary, it is the zero value function over the entire domain.

1.1.3 The variable coefficient Poisson's equation

As already mentioned, PDEs can describe real applications. The Poisson problem especially can have internal modifications by becoming the *variable coefficient Poisson's equation*, when the discontinuities and other characteristics can be described with an internal variable κ:

$$\nabla \cdot (\kappa(\boldsymbol{x})\nabla u(\boldsymbol{x})) = f(\boldsymbol{x}) \quad \text{on } \Omega \subset \mathbb{R}^d, \tag{1.21}$$

where $\kappa(\boldsymbol{x})$ can be constant anywhere, piecewise constant on internal sub-domains, or a function described in Ω.

The variable coefficient Poisson equation is part of the model problem proposed for the heat conduction equation which describes a phase-changing material. Given its relevance with this thesis some methods of resolution for this equation on complex geometries will be reported below.

About The Solution: Weak Formulation

We analyse in this part the weak formulation for the equation (1.21), with added internal conditions, introducing a boundary value problem that will be mentioned several times throughout this thesis.

We recall some space definitions useful in following

$$L^2(\Omega) = \{u : \Omega \to \mathbb{R} \,|\, \int_\Omega |u|^2 < \infty\}$$

$$(u, v)_{L^2(\Omega)} = \int_\Omega u(\boldsymbol{x})v(\boldsymbol{x})d\boldsymbol{x}$$

$H^s(\Omega) = \{v \in L^2(\Omega) \,|\, \partial_\alpha(\Omega) \in L^2(\Omega), \forall\alpha \in \mathbb{N}^n, |\alpha| = \alpha_1 + \ldots + \alpha_n \leq s\}$, $s \geq 0$

$\mathcal{D}(\Omega) = C_0^\infty(\Omega)$, the space of infinitely differentiable functions with compact support.

Given a solution for equation (1.21) with mixed boundary conditions (1.19)-(1.20), in [6], for piecewise constant κ the existence of Lagrangian and

punctual differentials is proved and their respective expressions are given. The author considers $g \in L^2(\partial\Omega_N)$, $\kappa(\boldsymbol{x})$ piecewise constant as κ_1 on Ω_1 and κ_2 on Ω_2, such that $\Omega_1 \cup \Omega_2 = \Omega$ and $\kappa_1, \kappa_2 > 0$. In this context a new computation for derivatives along discontinuities is proposed.

As above we consider the problem (1.21) with mixed boundary conditions (1.19)-(1.20), adding internal conditions through the domain discontinuity, and for a more general form of $\kappa(\boldsymbol{x}) > 0$, $\kappa(\boldsymbol{x}) \in L^\infty(\Omega)$:

$$
\begin{aligned}
-\nabla \cdot (\kappa(\boldsymbol{x}) \bigtriangledown u(\boldsymbol{x})) &= g(\boldsymbol{x}), && \text{in } \Omega_1 \cup \Omega_2, && \text{(1.22a)}\\
\kappa(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, && \text{on } \partial\Omega_N, && \text{(1.22b)}\\
u(\boldsymbol{x}) &= u_D(\boldsymbol{x}), && \text{on } \partial\Omega_D, && \text{(1.22c)}\\
[k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})] &= 0,\ R(k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S = [u(\boldsymbol{x})], && \text{on } \gamma, && \text{(1.22d)}
\end{aligned}
$$

with $\boldsymbol{n}$ the normal unit vector through $\gamma = \partial\bar{\Omega}_1 \cap \partial\bar{\Omega}_2$. R is a defined constant, in our case the *contact resistance* between two different materials. We invite the interested reader to see [7]-[8], where the results summed up in this section are not thorough enough.

We introduce the variational space

$$\mathbb{V} = \{v \in L^2(\Omega) : v_1 = v|_{\Omega_1} \in H^1(\Omega_1),\ v_2 = v|_{\Omega_2} \in H^1(\Omega_2)\},$$

with its broken norm and subspace

$$||v||_{\mathbb{V}} = (||v||^2_{H^1(\Omega_1)} + ||v||^2_{H^1(\Omega_2)})^{\frac{1}{2}}$$

$$\mathbb{V}_0 = \{v \in \mathbb{V} : v = 0 \text{ on } \partial\Omega_D\}.$$

Multiplying (1.22a) for a smooth function $v \in \mathbb{V}_0$ on both sub-domains, and integrating by parts, we obtain

$$\int_{\Omega_1} \kappa(\nabla u)(\boldsymbol{x}) \cdot (\nabla v)(\boldsymbol{x})\, d\boldsymbol{x} - \int_\gamma (\kappa\partial_{\boldsymbol{n}} u)(s) v(s)\, ds = \int_{\Omega_1} g(\boldsymbol{x}) v(\boldsymbol{x})\, d\boldsymbol{x},$$

$$\int_{\Omega_2} \kappa(\nabla u)(\boldsymbol{x}) \cdot (\nabla v)(\boldsymbol{x})\, d\boldsymbol{x} + \int_\gamma (\kappa\partial_{\boldsymbol{n}} u)(s) v(s)\, ds = \int_{\Omega_2} g(\boldsymbol{x}) v(\boldsymbol{x})\, d\boldsymbol{x},$$

with s as the tangential variables. Using (1.22d) and a sum of the above equations results in

$$\int_{\Omega_1 \cup \Omega_2} \kappa(\nabla u)(\boldsymbol{x}) \cdot (\nabla v)(\boldsymbol{x})\, d\boldsymbol{x} + \int_\gamma \frac{1}{R}[u](s)[v](s)\, ds = \int_{\Omega_1 \cup \Omega_2} g(\boldsymbol{x}) v(\boldsymbol{x})\, d\boldsymbol{x}, \tag{1.23}$$

resulting in the variational problem:

Find $u \in \mathbb{V}$, *such that*

$$u(\boldsymbol{x}) = u_D(\boldsymbol{x}) \tag{1.24}$$

on $\partial\Omega_D$ *and it satisfies* (1.23) $\forall v \in \mathbb{V}_0$.

Theorem 1.1.1 *Assume that the partition of* $\partial\Omega$ *into* $\partial\Omega_D$ *and* $\partial\Omega_N$ *is sufficiently smooth for* $\mathcal{D}(\bar{\Omega}_1 \backslash \partial\Omega_D)$ *to be dense in the space*

$$\{v \in H^1(\Omega_1) : v = 0 \text{ on } \partial\Omega_D\}.$$

Problems (1.22) *and* (1.23)-(1.24) *are equivalent, in the sense that any function in* $\mathbb{V}$ *is a solution of* (1.22) *in the distribution sense if and only if it is a solution of* (1.23)-(1.24).

We invite the reader to [7] for proofs and results on the variational problem. A study of norms and bounded spaces is proposed to obtain all tools needed to prove the well-posedness of the problem.

Theorem 1.1.2 *For any data* g *in* $L^2(\Omega)$ *and* $u_D \in H^{1/2}(\partial\Omega_D)$*, the problem* (1.23)-(1.24) *has a unique solution* $u \in \mathbb{V}$.

Moreover, a constant $c > 0$ exists such that this solution satisfies

$$||u||_{\mathbb{V}} \leq c(||g||_{L^2(\Omega)} + ||u_D||_{H^{1/2}(\partial\Omega_D)})$$

and a maximum discrete principle equivalence, used for existence in the general Poisson's problem, is given to complete the solution's existence.

1.1.4 Discrete Problem

Discretizations of an elliptic linear partial differential equation by finite differences, finite element or the finite volume methods result in a linear system to be solved. In general, the matrix induced by the numerical method is sparse, and it is necessary to use a good solver. Best cases involve symmetric positive-semidefinite matrices:

$$M^T = M, \boldsymbol{x}^T M \boldsymbol{x} \geq 0, \forall \boldsymbol{x} \in \Omega_h,$$

where Ω_h represents the discretized subspace of Ω. This difficulty creates a second problem, solving the discrete equation about the better solver to choose (among the direct and iterative ones). The immersed boundary methods use discrete delta functions on domain boundaries, allowing a flexible approach (see a.e. [9],[10]).

Triangular Meshes

A first refinement method is to approximate the surface by triangles (Fig.1.1) adjacent along edges. The discretization can stand on vertex points, centres or half-edges depending on the method.

On these kinds of meshes finite element (FEM) and finite volume schemes are often applied. The robustness of the method is sometimes related to the

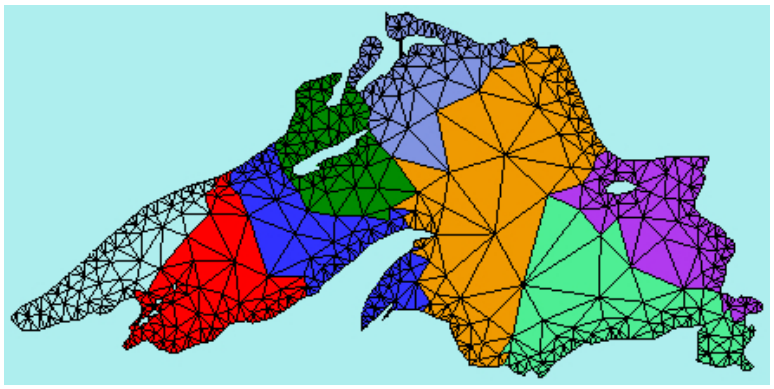


Figure 1.1: An example of a triangular mesh (*Triangle* product[0]).

geometry of these triangles and the dimension relationship between them when they are contiguous. Rules to triangularize, like the *Delaunay* property, are therefore required. Most of these discretizations entail sparse matrices; however, constraints for the mesh structure, like the Delaunay one, can ensure some advantages (symmetry, maximum principle, etc.).

Square Meshes

Quadtree and octree data structures have been proven to be optimal in such cases since they are able to capture the principal curvatures of a surface.

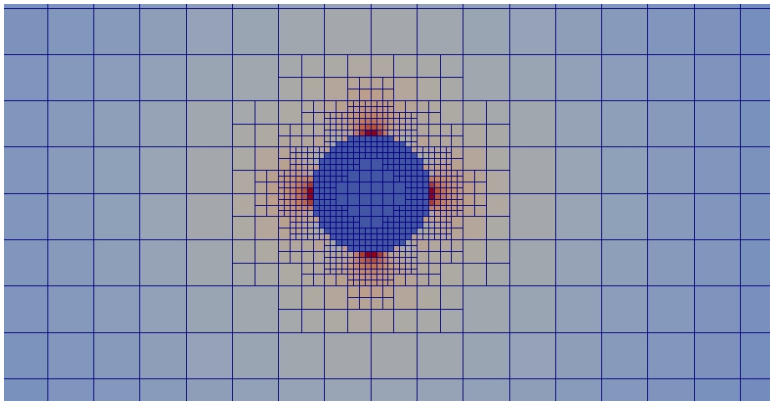


Figure 1.2: Graded grid on a cylindrical obstacle (current work).

The quad structures allow degrees of freedom on centres and nodes, simultaneously. In general, cell-centred methods involve symmetric matrices, since the neighbouring relationship is reflective. Depending on the problem to solve these meshes can afford different adaptive mesh refinement

[0] https://www.cs.cmu.edu/ quake/triangle.html

approaches: they can follow a discontinuous geometry at the beginning of computation or they can evolve in time. They can also be graded or unbalanced where required, and so on.

This advantage of being particularly compliant with the numerical problem has, over the years, led to an increase in the use of square meshes for numerical methods in the PDE resolution field.

1.2 Complex geometries approaches

1.2.1 Finite Difference Schemes

The use of a finite difference method (FDM) is nothing more than a direct conversion of the Poisson equation from continuous functions and operators into their discretely sampled counterparts. This converts the entire problem into a system of linear equations that may be readily solved via matrix inversion.

To solve (1.1) McCorquodale et al. [11] proposed a nodal finite difference scheme on rectangular Cartesian grids. In uniform areas, as well as we see for our method, they use standard discretization, while in jumping areas they consider the mesh as the coarser-level values as follows. For the generic point j on the jump boundary at level ℓ of refinement:

if the level-ℓ node j coincides with a node at level $\ell - 1$, the value is projected;

else an interpolation (quadratic in two dimensions and bi-quadratic in three dimensions) is applied to obtain the value on j.

Even if the truncation error is $O(h)$ on the interpolations. The solution error computed with this method is $O(h^2)$.

McKenney et al. [12] devised a robust fast solver with an integral part for Laplace equations on complicated domains; they used a finite difference operator on the Poisson equation with a discontinuous right hand side. Their solver is developed on rectangular embedded mesh and distinguishes *regular* and *irregular* points whether the discretization stencil is entirely contained in a discrete sub-domain part or not.

Gibou et al. [13] proposed a second-order accurate symmetric discretization for the variable coefficient Poisson equation (1.21) on irregular domains, taking into account jumps along internal surfaces. In this work, they use a level set formulation to represent the interface location and a finite difference discretization of the heat equation on a Cartesian grid to solve for the temperature. The level set method was developed in the 1980s by the American mathematicians Stanley Osher and James Sethian [14]-[15]; this method became popular in many mathematical applications.

In absence of irregular surfaces, for a Δ_x uniform Cartesian mesh size, the standard finite difference discretization for (1.21) is:

$$\frac{\kappa_{i+\frac{1}{2}}\left(\frac{u_{i+1}-u_i}{\Delta_x}\right) - \kappa_{i-\frac{1}{2}}\left(\frac{u_i-u_{i-1}}{\Delta_x}\right)}{\Delta_x} = f_i. \tag{1.25}$$

This formula can easily be extended in more dimensions; the symmetric matrix that this method produces is computationally inexpensive. For the method in this article, since a grid point is not contained in a subdomain, its ghost value is properly chosen (for u or κ where needed) using constant, linear and quadratic interpolation as appropriate. The choice of the better interpolation is a common problem when level jumps occur in most cases. A supra convergent finite difference method is given in [16], where the interpolation on non-graded grids are studied both in two and three dimensions. In [17], this node-based adaptive mesh refinement framework based on quadtree/octree Cartesian grids is applied for an approach to solve the Poisson problem, the diffusion equations, and the Stefan problem. A level-set approach is used to capture the complex geometry or the free boundary. The method is second-order (some cases fourth) accuracy for L_∞ norm.

1.2.2 Finite Element Schemes

The numerical efficiency of finite element methods (FEM) is mostly attributed to the fact that such formulations always yield symmetric linear systems, which are computationally inexpensive to invert; however, on AMR approaches the finite elements can be difficult to implement. On the other hand, the advantages of using Cartesian meshes are the simplification of data structure and the method of fluxes formulation handling.

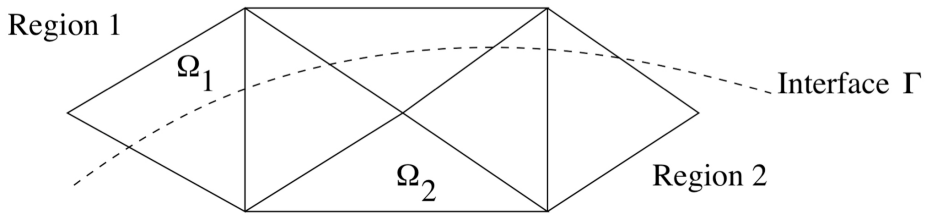


Figure 1.3: Non-conforming interface passing through unstructured (triangular) mesh [18].

Huh and Sethian [18] introduced a non-conforming finite element method, on triangular meshes, for second-order elliptic interface problems. Given a standard background mesh and an interface that passes between elements (Fig. 1.3), they devised a singular correction function that satisfies the jump conditions and provides accurate sub-grid resolution of the discontinuities.

Young et al.[19] introduced a finite element method employing adaptive mesh refinements for second-order variable-coefficient elliptic equations using a cut-cell representation of irregular domains. Their method uses box finite elements defined by a Cartesian, octree data structure grid that is independent of the boundary definition, and their discretization is non-linear.

1.2.3 Other Approaches

Johansen and Colella [20] treated the problem (1.21) on complex geometries, with $\kappa(\boldsymbol{x}) > 0$, using a finite volume scheme on Cartesian grids. Their method is cell centred on a rectangular grid; even if the rectangular centres lie outside the domain of interest, the irregular boundary geometry is represented locally by intersecting the domain Ω with each rectangular cell and approximating the operator using a conservative, finite volume discretization. This approach is combined with a standard five-points discretization on interior points. Even if the truncation error on boundary discontinuity is first order, they use a second-order accurate differences discretization on fluxes and they ensure overall a solution second-order accurate. This method allows them to solve a linear system and they can prove that it is well-conditioned; however on adaptive mesh refinement, an approach, similar to the one seen in [11], is applied to interpolate a value on a fine-level grid from the coarse grid (Fig. 1.4). The method proposed is a specific application of a general formalism for constructing consistent finite difference methods for problems with irregular boundaries, as the one proposed in the current work.

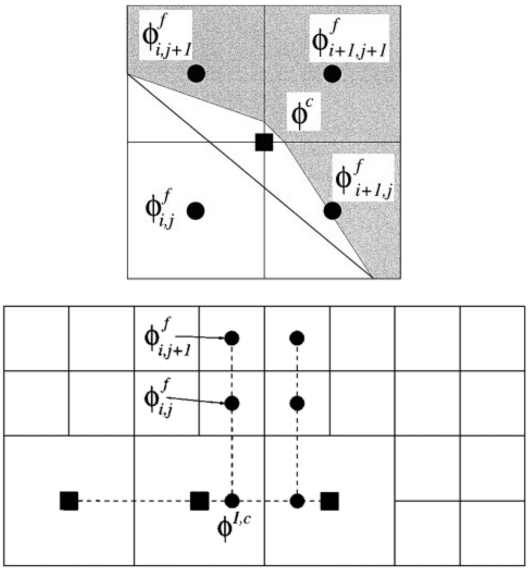


Figure 1.4: Coarse-fine stencil approach in [20]

On Cartesian-grids approaches, we also find in Marques et al. [21] that implemented an algorithm able to compute interfaces across which jump conditions are prescribed for both the solution and its weighted normal derivatives. Their algorithm is a combination of the correction function method (CFM) and boundary integral formulations of the Laplace equation, as presented in McKenney's strategy. Li et al. [22] presented a diffuse domain approach that provides numerical simulations using adaptive, multilevel finite-difference, and finite-element methods. Advantages of their method are that it can be easily combined with a diffuse interface approximation of PDE and that the geometry of the domain does not need to be given analytically; moreover, this diffuse domain approach does not require any modification of standard finite elements or finite differences.

The *Voronoi Interface Method* was presented for elliptic problems on irregular domains by Guittet et al. in [23],[24]. Given a uniform mesh, they modify the mesh so that the irregular interface coincides with the edges of the new mesh and the degrees of freedom close to the interface are all located at the same distance from the interface. The new structure will contain the Voronoi interfaces (Fig. 1.5), on which a finite volume approach is conceived, obtaining a first order accurate method on both uniform and adaptive octree base meshes in two and three dimensions.

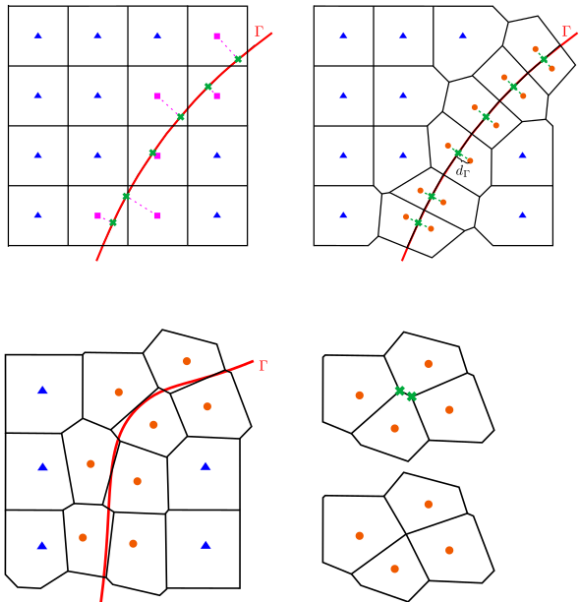


Figure 1.5: Illustration of the procedure for generating a Voronoi diagram based computational mesh [23].

1.2.4 Applications To Navier-Stokes Equations

The problem of PDEs on complex geometries include applications of the Navier-Stokes equations, where some methods reduce part of the resolution to solve the Poisson equation on irregular domains.

We consider non-dimensional incompressible Navier-Stokes equations (NSE) in a general and bounded domain Ω:

$$\partial_t \boldsymbol{u}(\boldsymbol{x}) + (\boldsymbol{u}(\boldsymbol{x}) \cdot \nabla)\boldsymbol{u}(\boldsymbol{x}) = -\nabla p + \nu \Delta \boldsymbol{u}(\boldsymbol{x}) \quad \forall \boldsymbol{x} \in \Omega, \tag{1.26}$$

$$\nabla \cdot \boldsymbol{u}(\boldsymbol{x}) = 0 \quad \forall \boldsymbol{x} \in \Omega. \tag{1.27}$$

For simplicity, from now on we will call $\boldsymbol{u}$ the term $\boldsymbol{u}(\boldsymbol{x})$. We apply the *Chorin's projection method* (see [25]) to (1.26), which computes an intermediate velocity u^* explicitly using the momentum equation by ignoring the pressure gradient term:

$$\frac{\boldsymbol{u}^* - \boldsymbol{u}^n}{\Delta t} = -(\boldsymbol{u}^n \cdot \nabla)\boldsymbol{u}^n + \nu \Delta \boldsymbol{u}^n, \tag{1.28}$$

where $\boldsymbol{u}^n$ is the velocity at the n-th time step, then the projection step corrects the intermediate velocity to obtain the solution at time $n+1$:

$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^* - \Delta t \nabla p^{n+1} \Leftrightarrow \frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^*}{\Delta t} = -\nabla p^{n+1}. \tag{1.29}$$

To compute the pressure term at time $n+1$, we use di divergence of (1.29), requiring that $\nabla \cdot \boldsymbol{u}^{n+1} = 0$ as in (1.27). We obtain:

$$\Delta p^{n+1} = \frac{\nabla \cdot \boldsymbol{u}^*}{\Delta t}. \tag{1.30}$$

Problem (1.30) is a Poisson problem. Thus, solving these internal pressures requires computational methods devised for the Poisson problem on complex geometries, such as above.

Losasso et al. [26] developed an octree data structure method where they used all dimensions of the mesh. The velocity components are defined on the cell faces, while the pressure is defined at the centre of the cell, and other data are stored at the nodes. Moreover, the implemented method is symmetric positive definite, enabling the use of fast solution methods. The value of a new node on an edge is defined as the average of its two neighbours, and the value of a new node at a face centre is defined as the average of the values on the four corners of that face. The velocities on the new faces are defined by first computing the velocities at the nodes and then averaging back to the face centres. Nodal velocities are computed by averaging the four values from the surrounding cell faces as long as the faces are all the same size. This method is first-order accurate for the Poisson solver part that stands on non graded grids.

Popinet [27] proposed a second-order non-symmetric numerical method to study the incompressible Navier–Stokes equations using an octree data structure. In his method, he solves a Poisson equation for pressure using a cell-centred approach; moreover, the interpolations involve adjacent cells using both an exact projection for face-centred advection velocities and an approximate projection for the final projection of the cell-centred velocities.

A fast finite difference method, on Cartesian grids, is given by Li and Wang [28]. Using the vorticity stream-function formulation to solve the Navier-Stokes equations, they solved the Poisson equation on irregular domains; thus, they applied a central finite difference scheme on immersed boundary approaches. As discussed before, they had to determine regular and irregular grid points using interpolations on interface intersections.

Fadlun et al. [29] developed a second-order accurate, highly efficient method for simulating unsteady three-dimensional incompressible flows in complex geometries. They used boundary body forces that allow the imposition of the boundary conditions on a given surface not coinciding with the computational grid. The scheme is an immersed boundary method combined with a second-order finite difference scheme.

Olshanskii et al. [30] introduced a finite difference solver for the unsteady incompressible Navier–Stokes equations based on adaptive Cartesian octree grids. They used graded mesh, treating the staggered location of velocity and pressure unknowns on cubic meshes. The pressure degrees of freedom are assigned to cells centres and velocity variables are located at cells faces. A similar cell-centred approach is applied on a parallel multigrid Poisson solver, developed by McAdams et al. [31], where a complete analysis of multigrid advantages in parallel are presented in detail.

1.3 Preliminary Conclusions

Several different numerical methods have been introduced in this chapter in order to provide a broad view of the various possible resolutions to the problem. In the broad set of these methods and their continuous evolution and improvement, the numerical method of this thesis is framed.

Some discretization problems can arise in AMR approaches when a level jump occurs in the numerical domain. The majority of these methods overcome this obstacle by using, as seen, internal interpolations that can approximate the value of a fictitious neighbour; these methods thus, introduce an interpolation error with the aim of optimizing it to higher orders, this to ensure the convergence of the finite difference method. We present a finite difference method adapted to local geometry in the presence of level jumps that does not makes use of internal interpolations: the error introduced by our method is related to the truncation error of the discretization. One reason for using the FDM is the easy conception that applies to discretiza-

tion, so our method is easy to understand besides being locally accurate; our variant aims at optimize the finite difference method in a context of an adaptive mesh refinement approach.

Chapter 2

The Octree Data Structure

At the centre of each numerical method for solving differential equations is the way in which we discretize a continuous domain of interest in a grid of many individual elements. This subdivision can be of the static type, when it is established once at the beginning of the calculation, or dynamic, when the grid itself progresses with the calculation. If you want to create a greater refinement in some areas compared to others, you need to create very dense static refinement or adopt dynamic methods. The **adaptive grids refinement** proposes to work on simple discrete spaces on the graphic plane. For this reason, they are represented by regular figures that are easy to manage. Therefore, we will work in detail with squares and cubes, each of which will be represented as a node of a quadtree and octree respectively, which, depending on the case, will be cleaved on some parts of the domain. In this thesis we analyse the duality of the grid, what is visually represented, and the tree, which is a useful computational tool for our purpose.

This chapter introduces the definitions needed to understand the structure, which are presented with more in-depth specifications for those applied in our code.

As explained before, following a discontinuity, or a complex geometry, requires accuracy in some parts of the numerical domain. To this purpose, a number of advantages of the structure are presented in general, and then a more thorough look at some of the qualities used in this work follows. In what follows we highlight, on one hand, the conceptual simplicity of the octrees in their spatial occupation, and on the other hand, the computational efforts that can determine choices during the construction of the mesh.

2.1 Quadtrees and Octrees Introduction

In graph theory, a tree is an undirected graph in which any two vertices are connected by exactly one path. In other words, any acyclic connected graph is a tree. However it is possible to define it as a data structure independent

of a sub-set family of graphs.

Definition 2.1.1 *A **tree** can be seen as a collection of objects of the same nature, called **cells** or **octants**, which are connected to each other in an appropriate manner by arches.*

Remark 2.1.1 *Other definitions can be attributed to this structure. For example:*

- *a tree is a data structure made up of nodes and edges without having any cycle.*
- *a tree is a non-linear data structure that organizes data in hierarchical structure and this is a recursive definition.*

The cells of a tree may contain various types of information; the important thing is that all the nodes of a tree itself are homogeneous: they contain the same type of information during computation.

Definition 2.1.2 *A **rooted tree** T is defined as a finite set of nodes labelled such that:*

- *there is a very specific node called **root**;*
- *the remaining set nodes are divided into N subtrees, respectively with roots $T_1, T_2 \dots T_N$, which, besides being connected at the root, they are mutually disjoint, i.e., they have no nodes in common.*

Definition 2.1.3 *When, for each generation of descendant nodes, N is set to four we will consider it to be a **quadree**. If it is fixed at eight, it will be considered an **octree**.*

In other words, a quadtree is a tree data structure in which each internal node has exactly four children. This data structure was named a quadtree by Raphael Finkel and J.L. Bentley in 1974 [32].

Definition 2.1.4 *A node directly connected to another node when moving away from the root is a **child**. The nodes of a tree with no children are called **leaves**. A node with children is named **father**.*

Definition 2.1.5 *Given the node P contained in a tree T, the **level** $\mathcal{L}(P)$ of node P is the whole number defined by recursion:*

- *if P root T then $\mathcal{L} = 0$;*
- *if P is not the root of T, let F be the father of P. Then $\mathcal{L}(P) = \mathcal{L}(F)+1$.*

The depth, height, or level of a tree T, $\mathcal{L}(T)$, is the whole number $\max_{P \in T}\{\mathcal{L}(P)\}$.

Definition 2.1.6 *Each leaf C has two kinds of* neighbours*: through faces following the axial directions and through corners following its diagonals directions. The mesh generated by a quadtree is defined as* ***graded (or balanced 2:1)*** *if:*

- *the levels of face neighbouring cells can not differ by more than one;*
- *the levels of diagonally neighbouring cells can not differ by more than two.*

There are several algorithms to manage a tree, and their efficiencies are often given by the type of application. For example, search and visit operations can be done from top to bottom, or from leaves to neighbourhood, by climbing levels, by using the node position in the graph, by the geometric position that it represents or by searching for the information it contains. The optimized method to access a trees informations is so an open problem depending on the application involved. For example, in [33], a traversal method for standard basic quadtree structures involves point and region location.

2.1.1 Common Applications

Quadtrees are used in many applications; the information they contain determines the type of structure. A particularly fruitful usage is in **image processing**. In this case, we can talk about *region quadtrees* (see [34],[35]), which represent a partition of space in two dimensions by decomposing the region into four equal quadrants, subquadrants, and so on, with each leaf node containing data corresponding to a specific subregion. The use of quadtrees to represent an image improves memory usage, especially for images where there are large areas of one colour. The recursion concept starts with an image in the form of a squared (2D) or cubical (3D) volume and recursively subdivides it into four/eight congruent disjoint octants until blocks of a uniform colour are obtained or a predetermined level of decomposition is reached [36]. This representation of the images gives ample scope for their union and intersection by allowing fluent algorithms for overlapping or comparing elements translated from pixels to octants (see Fig 2.1).

Another common use of quadtrees is in **search algorithms** field, in particular for representing multidimensional point data, where you can talk about *point quadtrees* [37]. Point quadtrees can also be used for metric problems, representing the different points of a map to determine the best links between distances. An example of this particular application is given by Samet and Hanan in [38], where an algorithm to track and measure all cities within 50 miles of Washington D.C. is presented.

The point quadtrees and the region quadtrees are similar in structure and decomposition strategy; a the substantial difference is the data contained.

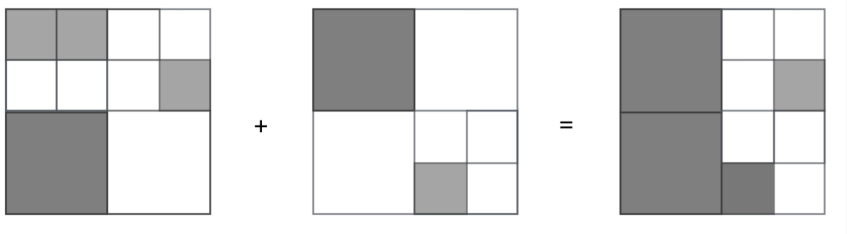


Figure 2.1: Quadree image representations: example of union.

Indeed some applications could also refer to point-region (PR) quadrees for this reason, as appropriate quadtrees can be interpreted differently. Some others categories exist, such as edge quadtrees, compressed quadtrees, and so on.

Among the various relevant applications of octrees is **mesh generation**, in sub-quads structure sometimes followed by a triangulation. The reason why quadtrees are functional, both in imaging and building numerical methods, is the relative simplicity with which it is possible to identify a quadtree with a spatial section that suits the need. Choosing the right data structure can significantly reduce the time complexity of a task. In general, adapting a mesh to optimize a problem is called **Adaptive Mesh Refinement (AMR)**, and it is in this sense that we make use of quadtrees in this work .

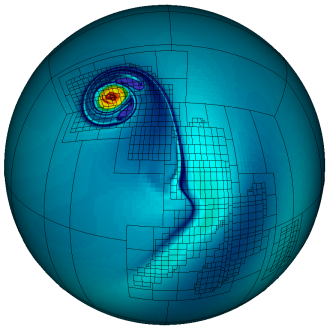
2.1.2 Adaptive Mesh Refinement Outlines

The AMR approaches have common purposes, if compared to uniform refinements. In particular the uniform meshes require high resolution for handling difficult regions (discontinuities, steep gradients, shocks, etc.), and they could be extremely computationally costly. To face this difficulty and to zoom in on regions of interest, adaptive mesh refinement has seen more and more use in computation applications.

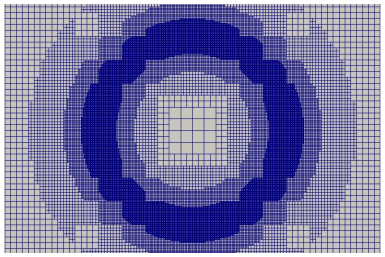
Different approaches exist for managing non-uniform meshes (Fig. 2.2). Completely unstructured AMR is identified by irregular connectivity. It cannot easily be expressed as a two-dimensional or three-dimensional array in computer memory; moreover, it provides greater geometric flexibility and fine-scale adaptivity at the cost of explicitly storing all neighbourhood relations between mesh elements.

Hierarchical hybrid grids split the domain into unstructured macroelements and they contain a mixture of structured portions and unstructured portions.

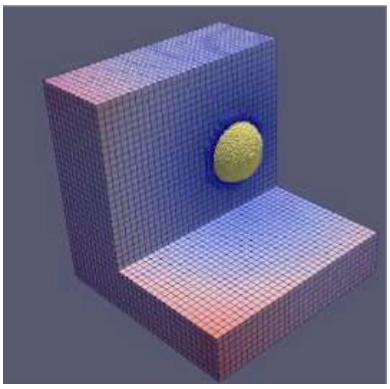
Block-structured AMR methods utilize unions of possibly mapped regular grids, which can be encoded cheaply. They are identified by regular connectivity. The possible element choices are quadrilateral in 2D and hexahedra in 3D. This model is highly space efficient, since the neighbourhood



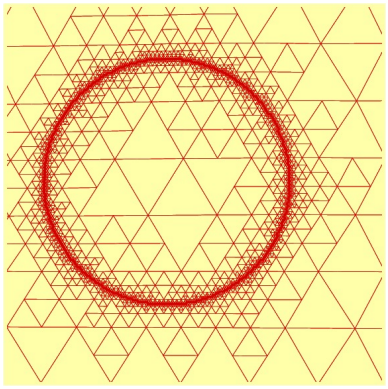
AMR on the cubed-sphere grid (McCorquodale et al. (2015), Chombo[a])



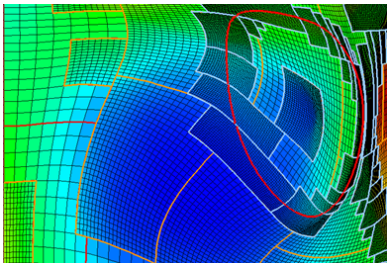
Quadtrees mesh refinement (current work, PABLO[b])



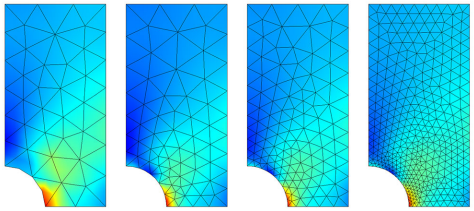
Octrees Mesh Refinement (Optimad, PABLO[b])



Triangular mesh refinement (Noureddine Hannoun[c])



Non-orthogonal moving mesh (NERSC)[d]



Finite element mesh refinement on a stressed point (FEA software, COMSOL[e])

Figure 2.2: Some examples of different AMR approaches. [abcde]

[a] http://www-personal.umich.edu/~cjablono/amr.html
[b] https://github.com/optimad/bitpit
[c] http://perso.usthb.dz/~nhannoun/research.html
[d] http://www.nersc.gov/about/nersc-staff/application-performance/alice-koniges/ale-amr/
[e] https://www.comsol.com/multiphysics/mesh-refinement

relationships are defined by storage arrangement.

Some advantages of AMR methods are shared irrespective of the type of structure applied, such as:

- starting with a coarse grid and identifying regions that need finer resolution;
- superimposing finer sub-grids only on those regions that increase computational (storage and time) savings over a static grid approach;
- compared to the fixed resolution of a static grid approach, this method offers accurate calculations on the zones of interest.

Recently, the use of quadrees and octrees for dynamic refinement methods has become increasingly common, in particular for solving PDEs: Balaras and Valella [39] proposed adaptive mesh refinement for immersed boundary methods based on a second-order finite difference scheme, obtained by ghost-cells interpolations; Min et al. [16] proposed a finite difference application for the variable coefficient Poisson equation on non-graded grids using interpolated values along jumps; Popinet [27] conceived of a finite volume, tree-based solver for the incompressible Euler equations in complex geometries. His method is second-order and uses a standard projection method.

One of the reasons for choosing quadtrees is the easy visual interpretation of these structures; the connectivity of tree structures also provides the information needed on multigrid methods. In this thesis, for simplicity, the quadtree is used as an example in many cases, specifying that similar reasoning can be used for octrees.

2.2 Ordering

An obvious problem with using AMR approaches can be seen in the way these meshes are ordered and more complex than a uniform mesh, where indexing of the elements is quite obvious.

Figure 2.3 shows the initial construction of a quadtree that follows an interface smaller than the domain that contains it. The tree root node represents the entire domain and is split in four parts to the next generation. As the refinement continues towards the sub-domain, the leaves, highlighted in red in the image, represent a square partition independent of the others.

Level 3 in Figure 2.3 is, in particular, an unbalanced tree. In this thesis, we use a particular type of tree structure, called the **Linear Octree**, in which only tree leaves are stored in memory (red dots in the figure). This type of data structure has evident advantages in memory gain and apparent difficulties in mesh management, such as sorting. Linear octrees are part of a wider category of quadtrees called compressed quadtrees, designed precisely

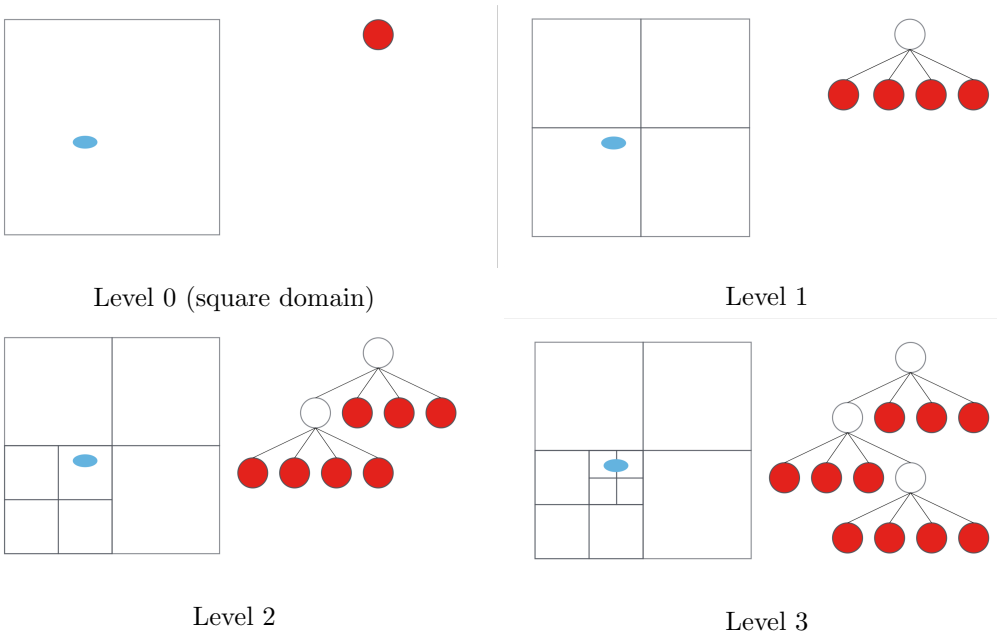


Figure 2.3: Construction of a quadtree for a square domain with an internal subdomain.

for memory savings and the selection of relevant sub-trees from the full structure (see [34]).

The example shown in Figure 2.3 reinterprets an example of Campbell et al. [40], in which special attention is given to the concept of **space-filling curves**; these curves are necessary to sort the elements of this type of mesh and possess the ability to cross every space partition once, thus they automatically generate a one-dimensional continuous mapping of the tree representation. Space-filling curves, therefore, address the problem of ordering the AMR octree structures, and several may exist if they comply with the aforementioned properties [41].

We briefly present the most well-known filling curves, and we focus on the last one that is applied in this thesis. In general, the mapping should preserve proximity in the sense that neighbouring cells should ideally correspond to adjacent blocks on the domain. The presented orderings follow the common idea of an initial curve that is pursued and imitated in the following levels. Sometimes these curves can resemble each other, and the distinction is the linking or sequence algorithm of the octants that are crossed by the filling curves such as the Hilbert filling curve (figure 2.4), which is derived from a different indexing algorithm of the more general Gray code filling curves (figure 2.5), where adjacent quadrants differ only by one bit in a binary representation.

The Hilbert filling curve consists of a sequence of iteratively defined

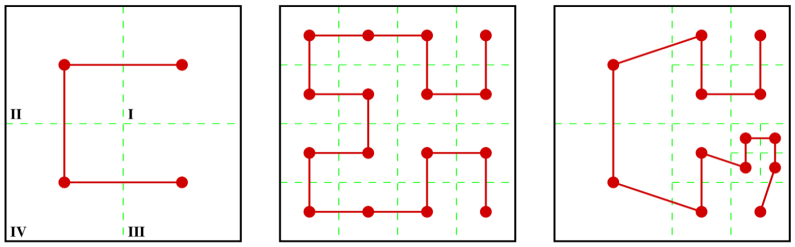


Figure 2.4: Hilbert filling curve ordering on quadree data structure [40].

curves. This space-filling curve, in image-mapping applications, offers great locality, as it exploits the correlation between pixels within small regions; thus, it can be preferred in certain scenarios.

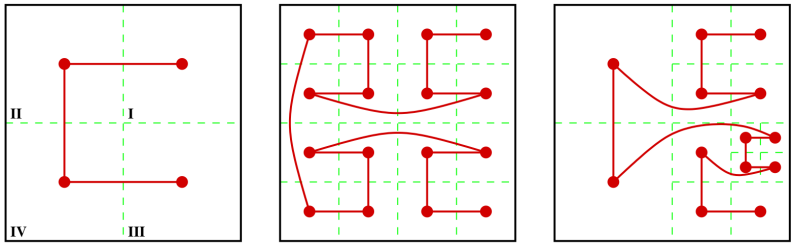


Figure 2.5: Gray code filling curve ordering on quadtree data structure [40].

The Morton ordering or Z-code (Figures 2.6, 2.8), is a space filling curve that follows an initial direction along a Z-like pattern.

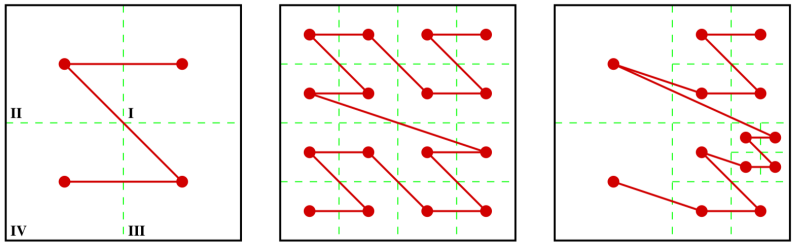


Figure 2.6: Morton filling curve ordering on quadtree data structure [40].

In this work, we refer to the order for local and global indices, as well as the search for neighbours along an octant's directions, using this space-

filling curve. An indexing of faces and vertices in the two-dimensional case, when we refer to the structure, is given in Figure 2.7.

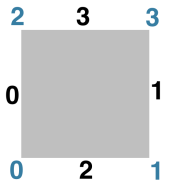


Figure 2.7: Internal Morton code of an octant through faces and vertices.

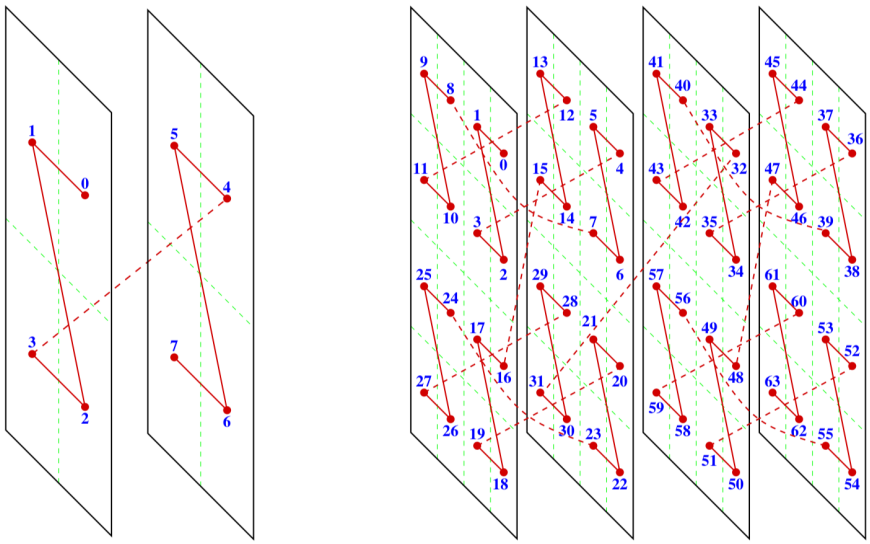


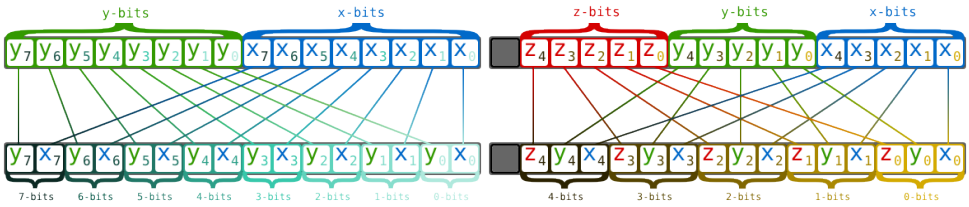
Figure 2.8: Morton filling curve ordering on octree data structure [40].

2.2.1 The Morton Code

The Morton code has been introduced in 1966 by G. M. Morton in *A computer Oriented Geodetic Data Base and a New Technique in File Sequencing* [42]. The aim of this work was to store, in sequence, a large amount of data concerning a wide geographical area. In particular, he focused on the Canada surface to store information about lands properties (inhabited areas, vegetation, etc.) with the purpose to be able to reuse this information for local development.

The algorithm to list the elements and determine the neighbourhood

with easy access to data starts from the position of an octant, using a point to determine it, with the information (x, y, z). The spatial coordinates of the point in the mesh are translated in binary, then the Morton code consists on interleaving the binary identifications. To conclude, we take the fractional part of each coordinate and expand it by inserting two "gaps" after each bit; then, we interleave the bits of all three coordinates together to form a single binary number (see Fig.2.9).



(a) 2D bits interleaving. (b) 3D bits interleaving.

Figure 2.9: Creation of Morton index by binary interleaving.[1]

The images in this section are presented in the documentation of the library Ash C++ [1]. To give an example of the process of creating the Morton index and binary construction, consider the Level 1 quadtree in Figure 2.10. [1]

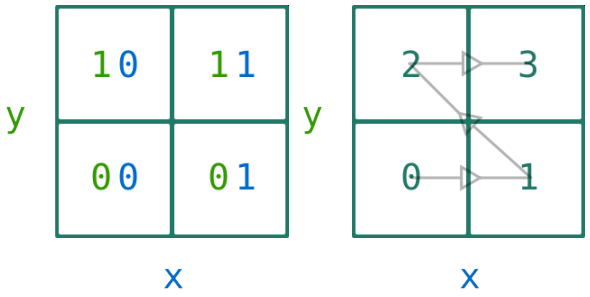


Figure 2.10: Computing Morton Code: step one of three.[1]

When the tree is broken into four children, the inner numbering of each quadrant follows that of the upper (2.11). By combining the two tracking indices, the ordered position is obtained for the global structure (2.12). We remark that a single level requires a 2-bit address and so on until leaves.

Spatial locality is an important factor in optimising access time to data elements. This means that accessing a data element in memory, then accessing another data element in memory that is nearby (has an address that is close to the first), can be cheaper by several orders of magnitude that accessing a data element that is far away. Standard c arrays (and many other

[1]Ash C++ Template Library (AshTL), `http://ashtl.sourceforge.net/index.html`

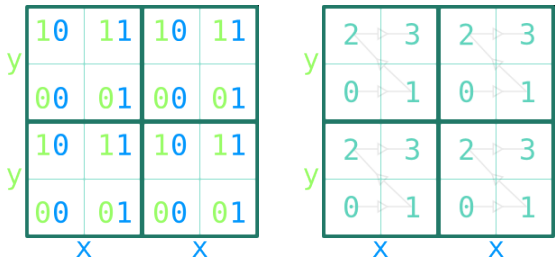


Figure 2.11: Computing Morton Code: step two of three[1]

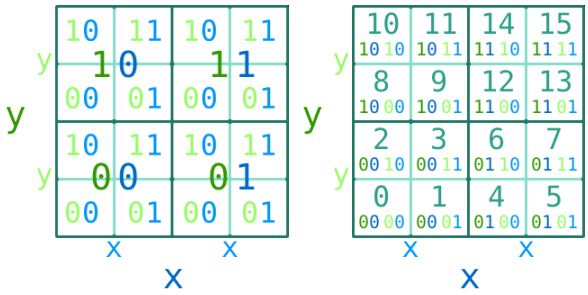


Figure 2.12: Computing Morton Code: step three of three [1]

data structures) have this property. This obvious advantage in looking for the neighbours of an octant, based on the numerical array that identifies it, is one of the reasons Morton code is preferred in this work.

The filling curve thus makes the balancing of the parallelisation topologically contiguous, advantage already stated but less obvious, easier to understand if the subdivision between the processors follows this sorting.

The structure management, the parallelisation, and its filling curves are made more efficient in this thesis thanks to the use of the PABLO library[2]; this library can efficiently manage both the Z-order of the elements and their local and global indexing, as well as allow processor communication that is almost automated in the library and therefore easy to access for the user. The Morton indexing used in PABLO manages the (x, y, z) coordinates for the octants (in two dimensions with value zero for z), respecting the principle presented above.

We can now fully summarize the type of structure that is referred to in this thesis work: *linear octree, ordered according to Morton code and optimized in parallel.* Our study focuses on graded meshes; however, we will present cases that are unbalanced to confirm the stability of the method.

[2]Optimad, PABLO `https://github.com/optimad/bitpit`, `https://optimad.github.io/bitpit`

2.2.2 Conclusions about the indexing in the code

In general, space-filling curves allow one to reduce higher-dimensional proximity problems, e.g., nearest neighbour search, to a one-dimensional problem. Solving such a problem typically involves searching and sorting in one-dimensional space. Using space-filling curves, these algorithms can be applied to higher-dimensional proximity problems. A further advantage of space-filling curves like the Hilbert curve is their recursive nature, which allows for them to be used for hierarchical indexing of higher-dimensional data. However the Hilbert curve is only suited for applications where a short path through the points is needed.

To conclude this part, we present four examples, devised by Campbell et al. in [40] (Fig.2.13), that study the partition quality of the three curves presented, examining the surface index versus the number of processes. The surface indices measure the interprocess communication. Let N_P be the total number of processes P_i, b_i be the number of partition boundary faces (elements faces that lie between two contiguous processes) and f_i be the total number of faces of P_i. We recall the definitions of *maximum local surface index* and *average local surface index* as:

$$r_M = \max_{i=1,\ldots,N_P} \frac{b_i}{f_i},$$

$$r_A = \frac{1}{N_P} \sum_{i=1}^{N_P} \frac{b_i}{f_i}.$$

Measuring the communication to compare the filling curves, we can distinguish *interprocess connectivity* as the percentage of other processes with which each process must communicate and *intraprocess connectivity* as the number of disjoint regions assigned to a given process. This value in particular can adversely affect the performance of some problems.

The comparison in figure 2.13 appears to favour the Gray or Morton codes for some tests and the Hilbert filling curve for most part. In the cases that seem to prefer Hilbert code, the mapping and inverse mapping processes are considerably more complex when compared to Morton code.

Z-order is mainly used to store data for fast searches due to the good locality it offers. The break due to jumps and connectivity that promote the Hilbert filling curves are facilitated, in our case, by internal communications handled by PABLO. Another reason to use the Morton code in our applications, in addiction to the good results for Hilbert curve, is that its computational cost is constant (and cheap) to access any point of the curve directly. Other curves may require recursive steps for their computation and result in bigger cost than a Z-curve with a minimal gain in terms of locality coherence.

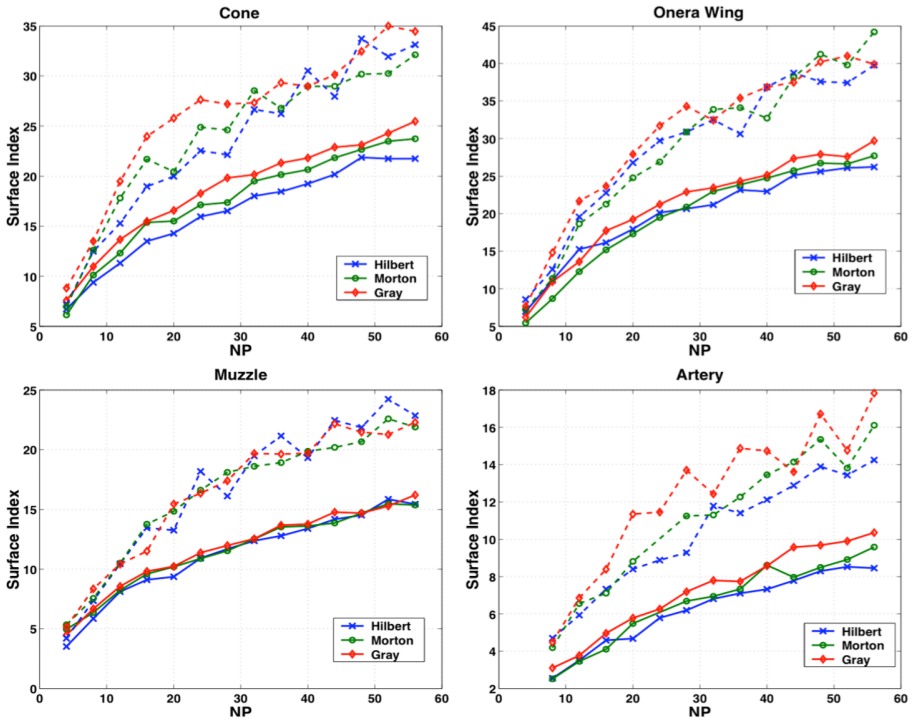
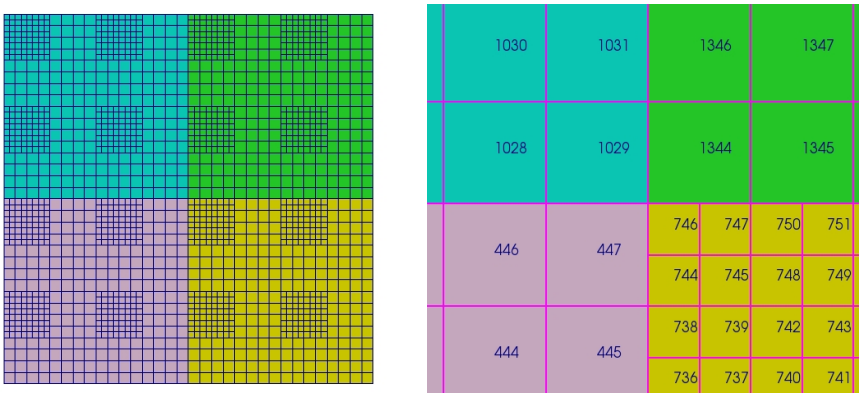


Figure 2.13: Surface indices vs. number of processes (N_P), four tests presented in [40]. Solid lines show the average surface index while the dashed lines show the maximum surface index.

The programming language used in this work was C++, and we mainly used two open-source external libraries: PABLO, mentioned above, which is a module of *bitpit*, for mesh management, and PETSc[3], for parallel calculation of linear systems.

The code has been implemented to be parallelly optimized. The first thing that can be highlighted in Figure 2.14 is whether the Z-order indexing managed with PABLO guarantees spatial contiguity where possible (2.14a), or following the order where it is not possible (isolated zones in Figure 2.16); on the other hand there is not contiguity in the global indexing managed with PETSC (2.14b). This mismatch causes sparsity of the numerical operator globally managed with PETSc that cannot be treated like a symmetric diagonal operator (see Fig. 2.15).

[3]PETSc home page `http://www.mcs.anl.gov/petsc/`



(a) (b)

Figure 2.14: A parallel partition of a structured grid and its global indexing along the cores intersections.

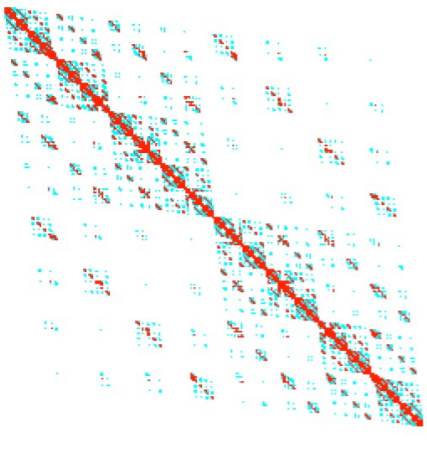


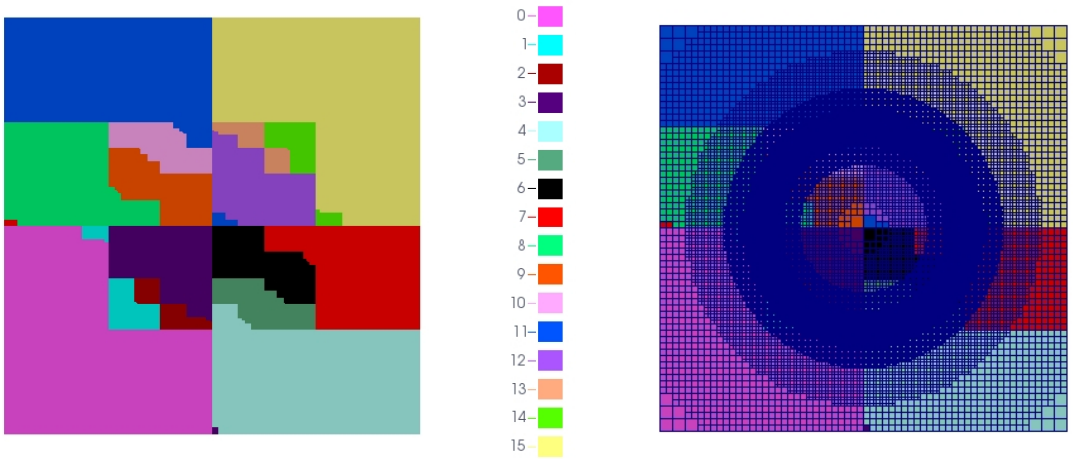
Figure 2.15: 3D PETSc operator matrix draw. 36128 points, octree level 6.

2.3 Octree mesh management libraries

We presented our rationale for choose PABLO in this work. However, octrees' mesh structures have recently obtained importance for their ability to solve numerical problems. Others libraries to approach these cases are in continuous evolution. We list some of these useful libraries, some of which are already associated with an appropriate resolution type (like finite element methods and go on).

Tao Chen devised **CAMINO**[4] (Cardinal's Advanced Mesh INnovation

[4] CAMINO web page `http://www-tcad.stanford.edu/tcad/bios/tchen.html#`



(a)

(b)

Figure 2.16: A parallel partition of a structured grid on 16 cores, the number of cells distributed present contiguous isolated coloured zones, following the global indexing.

with Octree), a 2D/3D octree mesh program that allows for easier tetrahedralization afterwards.

Point Cloud Library (PCL)[5]provides efficient methods for creating a hierarchical tree data structure from point cloud data; furthermore, it handles efficient nearest neighbour search routines.

DENDRO[6] is a collection of tools for parallel octree-based applications that supports PETSc objects. Similar to PABLO, this library is among the first to safeguard memory saving only the leaves of the tree's structure.

The **etree**[7] library is written in C and provides a couple dozen functions for creating, modifying, and searching octrees, including efficient mechanisms for appending octants and iterating over octants in Z-order.

The best performing library in this area, as far as we know, and for which we take some additional specification by comparing it to our choice, is **p4est**[8]. This library is designed to work in parallel and scales to hundreds of thousands of processor cores, adapted to AMR approaches [43]. Its most interesting characteristic is the ability to connect *multiple* adaptive octrees

page$\%$20a

[5] PCL web page `http://docs.pointclouds.org/trunk/group__octree.html`

[6] DENDRO github page `https://github.com/hsundar/dendro`

[7] etree web page `http://www.cs.cmu.edu/~euclid/`

[8] p4est web page `http://www.p4est.org`

into a forest of octrees that can represent a wide variety of geometric shapes. Some similarities to PABLO of this library are:

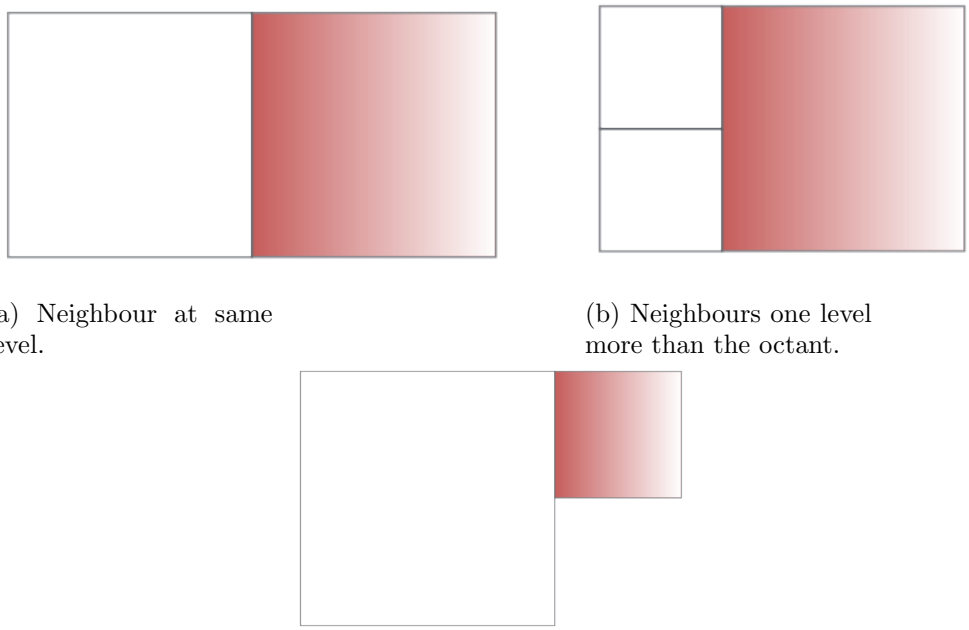
- the Morton code indexing of elements (the global indexing can across different trees);
- the balancing 2:1 relation is handled by the library when required;
- the parallel partition is user defined with a possible *wheighted partition*, which allows distributing the octants on the processors not according to the number but to a weighted relevance in the calculation.

Despite of several advantages of p4est, we refer in this section to PABLO as a module of bitpit, but we focus on its advantages, excluding its (more complex) container properties. Both libraries are open source, but PABLO's routines allow immediate inclusion within a code without needing important adaptation; this characteristic is useful for a user who can easily integrate an AMR approach from pre-existing code without completely rebuilding his strategy on classes bound to the management library. Moreover, PABLO is able to handle intersections, to balance the elements by refinement level strategies, and to balance 2:1 relations also along vertexes and edges, on the other hand, PABLO cannot manage more flanked trees at this moment. Furthermore, the main reason for our choice is that PABLO manages linear octrees. This allows a relevant gain of memory; in fact it counts approx. 30B per octant in 3D.

2.4 Identifying a Neighbourhood Configuration

We classify the neighbours topology of an octant using a base-5 8-digits numerical key (resp. 26-digits for the 3D case). To describe the possible set of contiguous neighbours we need five values whose choice is defined in detail below. Four digits are devoted to face-adjacent neighbours, the remaining four to corners ones. Each octant looks at its neighbours following the internal index due to the Morton code (Fig. 2.7); we suppose that an octant analyses its neighbourhood through face indexed by zero (red cell in Figure 2.17), three cases are possible for a graded grid:

- The neighbour has the same level of refinement (Fig. 2.17a). In this case the two cells have equal dimensions and properties.
- The neighbours have a level of refinement more than the octant (Fig. 2.17b). Along the face of the octant two adjacent neighbours lie.
- The neighbour have a level of refinement less than the octant (Fig. 2.17c). This case implies the absence of at least a corner neighbour, for the case in figure is the corner indexed at zero (bottom-left one).



(a) Neighbour at same level.

(b) Neighbours one level more than the octant.

(c) Neighbour one level less than the octant.

Figure 2.17: Possible neighbourhood cases through a face.

The analysis done for the first face is repeated through the others and the corners. This study of neighbourhood allows us to conceive a tool representing the topology of an octant as follows.

We define a function of level: $[\mathcal{L}] := \mathcal{L}_o - \mathcal{L}_n$, with $\mathcal{L}_o$ the level of the concerned octant and $\mathcal{L}_n$ the level of the neighbour, so that the value attributed to the key elements are:

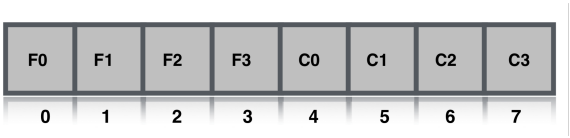
0	$\nexists$ neighbours on this side
1	$[\mathcal{L}] = 0$
2	$[\mathcal{L}] = -1$
3	$[\mathcal{L}] = 1$
4	$[\mathcal{L}] = -2$
5	$[\mathcal{L}] = 2$

This key has the following properties:

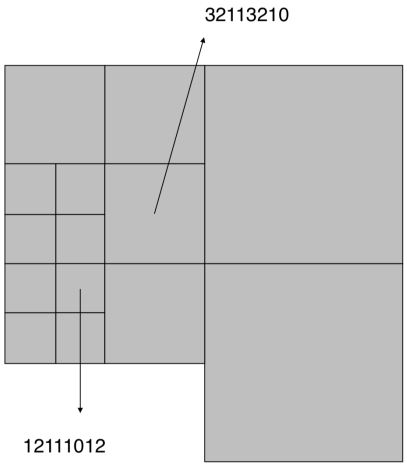
- it is bijective;
- it is elementary to build and clear to interpret;
- it is independent of the involved cell dimensions, i.e., the tree level.

This key is not strictly necessary, but it can lead to a significant speed up in the pre-processing phase. For this reason, often within the code, this tool will be used, even if not specified.

Figure 2.18 briefly shows the order of attribution of the function (2.18a) and the construction of the numerical sequence for two configurations (2.18b).



(a) 2D key order attribution, where F stands for face and C for corner.



(b) Example of key's generating.

Figure 2.18: Identifying a configuration.

To get an idea of how many configurations would have existed, taking only the balanced 2:1 case through the faces, a random production code of octrees is repeatedly launched; for each configuration, their respective identification keys have been saved. The count of the 2D configurations stabilized at 809, while in the three-dimensional case is over a million.

2.5 Preliminary Conclusions

In this chapter we introduced the octrees and we focused on space filling curves. Our code is natively designed in parallel, allowing us to overcome some of the indexing-related obstacles and also the benefits of using other curves, using external libraries properly. Moreover, the bijective identification of local topology in the case of graded grids gives us an instrument

for further advantages in terms of computing time. The octree-based solver is designed to be immediate for the user and therefore easy to handle and adaptable to different numerical and physical applications besides what is considered in this work. Lastly, we optimize the space filling curve applied to maximize its advantage, limiting the intra-process communication.

Chapter 3

The Finite Difference Method Computation

This chapter introduces the finite difference methods to approximate partial differential equations. The first part presents common properties of finite difference schemes: the analytical derivation, the convergence study and the advantages. The accuracy of these methods is directly related to the ability of a mesh to approximate a continuous system, and errors may be arbitrarily reduced by simply increasing the number of degrees of freedom. Then, a survey of their adaptation to hierarchical meshes is done; in this context, our method is included. We follow up the construction of the cell centred scheme applied in this thesis project and highlight the benefits of a method strongly dependent on the local geometries.

3.1 Introduction to Finite Difference Methods for PDEs

There are several methods for approximating a derivative on a point of a mesh. The finite difference method (FDM) works by replacing the region over which the independent variables in the PDE could be approximated with a neighbouring value [44]. This method is based on the use of Taylor's theorem, here enunciated, to obtain the approximate values.

Theorem 3.1.1 (Taylor's theorem) *Let u be an n-times differentiable function on an open interval containing the points a and x, and $h = (x - a)$ then:*

$$u(x) = u(a) + h \cdot u'(a) + \frac{h^2}{2!} \cdot u^2(a) + \frac{h^3}{3!} \cdot u^{(3)}(a) \ldots + \frac{h^{n-1}}{(n-1)!} \cdot u^{(n-1)}(a) + O(h^n), \tag{3.1}$$

where $O(h^n) = \frac{h^n}{n!} \cdot u^{(n)}(c)$ for some value c between x and a.

In calculus, Taylor's theorem gives an approximation of an n-times differentiable function around a given point by a truncated part of above formulation.

Definition 3.1.1 *Let a_r be the value $\frac{u^{(r)}(a)}{r!}$. The function T_n defined by*

$$T_n(x) = a_0 + a_1 \cdot h + a_2 \cdot h^2 + \ldots + a_n \cdot h^n$$

is called Taylor polynomial *of degree n of u at a and it can be thought as a polynomial which approximates the function f in some interval containing a.*

For analytic functions the Taylor polynomials at a given point are finite-order truncations of its Taylor series, which completely determines the function in some neighbourhood of the point. The error of this approximation is on the limit of the term $O(h^n)$, defined as **truncation error**. In general, when we discard this term, we get an approximation of $u(x)$ of error $O(h^n)$. The truncation error is often included in the more general definition of *discretization error*.

Following (3.1) and truncating after the first derivative, we obtain:

$$u(x) = u(a) + h \cdot u'(a) + O(h^2) \tag{3.2a}$$

$$\Rightarrow u'(a) = \frac{u(x) - u(a)}{h} + \frac{O(h^2)}{h} \tag{3.2b}$$

$$\Rightarrow u'(a) = \frac{u(x) - u(a)}{h} + O(h). \tag{3.2c}$$

The (3.2c) is called a **first-order** finite-difference approximation of $u'(a)$ since $O(h)$ depends on the first power of h. In general, when the Taylor development is truncated after ulterior derivatives, such as the truncation error of (3.1) depends on the $n+1$ power of h ,the finite difference method is of **order n**. h is called **step size**, and the formula (3.2c) is called a **forward** finite-difference approximation if $h > 0$, otherwise **backward**.

The tests in this work are at least of dimension two, and analogous formulae stand for further dimensions. We present the general bi-dimensional formulation. Let us consider when h_x and h_y are not necessarily equal and a function $u(x, y) : \Omega \to \mathbb{R}$, with $\Omega \subset \mathbb{R}^2$ and $u \in C^{n+1}(\Omega)$ derivable at least $(n + 1)$ times. The general two-dimensional Taylor polynomial formula of n-th order has the form:

$$u(x + h_x, y + h_y) = \sum_{0 \leq j+k \leq n} \frac{1}{j!k!} \frac{\partial^{j+k} u(x, y)}{\partial x^j \partial y^k} h_x^j h_y^k + E_{n+1},$$

where E_{n+1} stands for a truncation error of order $n + 1$.

3.1.1 The five-points difference operator

Let us consider the Poisson problem:

$$\Delta u(x,y) = f(x,y) \quad (x,y) \in \Omega, \tag{3.3}$$

$$u(x,y) = 0 \quad (x,y) \in \partial\Omega,$$

with $\Omega = [0,1] \times [0,1]$ the unit square in $\mathbb{R}^2$. For sake of simplicity we consider the homogeneous Dirichlet boundary condition. Let N be a positive integer and $h = \frac{1}{N}$. We consider the discrete cell-centred uniform mesh (Fig.3.1) in $\mathbb{R}^2$, described as:

$$\mathcal{M}_N = \{(ih + \frac{h}{2}, jh + \frac{h}{2}), i,j \in \{0,\ldots,N-1\}\}. \tag{3.4}$$

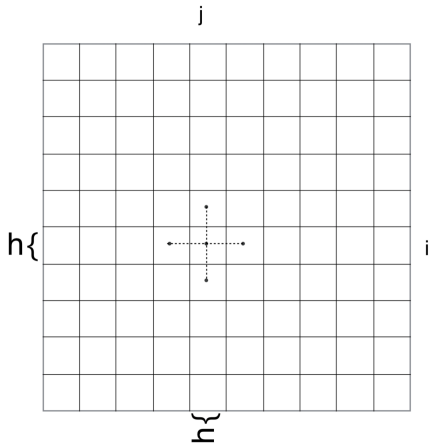


Figure 3.1: Uniform mesh. $N = 10$

To discretize (3.3), we use a discrete function $u_h : \mathcal{M}_N \to \mathbb{R}$ satisfying:

$$\Delta_h u_h = f_h \text{ in } \mathcal{M}_N, \tag{3.5}$$

where Δ_h is an operator on the discrete domain formulated on the mesh, and f_h the discrete values of the right hand side over grid points. We consider, for simplicity, the generic point u_{ij}, which lies in the inner part of the numerical domain, excluding the boundary; the natural choice is to see four neighbours for the concerned point: one each to the left, right, above, and below. Equation (3.1) following the two axes in both directions, with respect to u_{ij} gives:

$$u_{i-1,j} = u_{i,j} + h\partial_x u_{i,j} + \frac{h^2}{2}\partial_{xx} u_{i,j} + O(h^3), \tag{3.6a}$$

$$u_{i+1,j} = u_{i,j} - h\partial_x u_{i,j} + \frac{h^2}{2}\partial_{xx} u_{i,j} - O(h^3), \tag{3.6b}$$

$$u_{i,j-1} = u_{i,j} + h\partial_y u_{i,j} + \frac{h^2}{2}\partial_{yy} u_{i,j} + O(h^3), \tag{3.6c}$$

$$u_{i,j+1} = u_{i,j} - h\partial_y u_{i,j} + \frac{h^2}{2}\partial_{yy} u_{i,j} - O(h^3). \tag{3.6d}$$

A simple sum of which generates an approximation with respect of all considered neighbours equally considered in both axial directions.

$$(3.6a) + (3.6b) + (3.6c) + (3.6d)$$
$$\Downarrow$$
$$u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} = 4u_{ij} + h^2\partial_{xx} u_{i,j} + h^2\partial_{yy} u_{i,j} + O(h^3)$$

We remember the numerical equivalence of discretized operator combining the equation above as

$$(\Delta_h \approx \Delta = \partial_{xx} u + \partial_{yy} u)$$
$$\Downarrow$$

$$\Delta_h u_{ij} = \frac{u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{ij}}{h^2} + O(h^2). \tag{3.7}$$

This formulation is valid $\forall i, j \in \{1, \ldots, N-2\}$, there are several approaches for boundary points that we will see later. Equation (3.7) constitutes the five-points stencil for the Laplacian approximation, and it satisfies particular properties [44]-[45]:

Theorem 3.1.2 *If $u \in C^2(\bar{\Omega})$, then*

$$\lim_{h\to 0} ||\Delta_h u - \Delta u||_\infty = 0.$$

If $u \in C^4(\bar{\Omega})$, then

$$\lim_{h\to 0} ||\Delta_h u - \Delta u||_\infty \leq \frac{h^2}{6}\sigma,$$

where $\sigma = \max\left\{\frac{\partial^4 u}{\partial x^4}, \frac{\partial^4 u}{\partial y^4}\right\}$

The matrix of this scheme has special advantages:

- it is sparse, with at most five elements per row non-zero;
- it is block tridiagonal;
- it is symmetric;
- it is negative definite;

- its diagonal elements are negative, the others are positive.

The problem (3.5) could be rewritten, on interior points, in matrix form as $Au_i = f_i$, with

$$A = \begin{pmatrix} B & I & O & \dots & O \\ I & B & I & \dots & O \\ O & I & B & \dots & O \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ O & O & O & \dots & B \end{pmatrix},$$

$B = \begin{pmatrix} -4 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & -4 \end{pmatrix}$, I the $\mathbb{R}^{3\times 3}$ identity matrix and O the $\mathbb{R}^{3\times 3}$ zero matrix. We will see in the following, studying the finite difference method of this work, that the regularity of the matrix describing the operator is not taken for granted. It is sufficient to think about the Z-order indexing and the global matrix previously presented in Chapter 2, Fig.2.15.

3.1.2 Boundary Conditions

We discuss the finite difference strategies to impose Dirichlet and Neumann boundary conditions, which concern this work; however, other methods and cases can be found in [46]-[47]. For simplicity, we consider the one-dimensional case; the same reasoning is valid for further dimensions.

$$\Delta u(x) = f(x), \quad x \in \Omega = [0, 1],$$

where the unit interval is subdivided in n points uniformly $h = \frac{1}{n}$.

Dirichlet boundary conditions

Given Dirichlet boundary conditions

$$u(0) = a,\ u(1) = b,$$

we define the interior unknowns vector $U = \{u_1, \dots, u_{n-2}\}$ and the complete set $\bar{U} = \{u_0, \dots, u_{n-1}\}$. It is clear that the boundary conditions resulting from the continuous problem are

$$u_0 = u(x_0) = u(0) = a,\ u_{n-1} = u(x_{n-1}) = u(1) = b. \tag{3.8}$$

We present two simple ways to deal with the discrete problem $\Delta_h u_h = f_h$.

First approach

In general, $\forall j \in \{1, \ldots, n-2\}$ stands the discretization

$$\frac{1}{h^2}(u_{j-1} - 2u_j + u_{j+1}) = f_j, \tag{3.9}$$

where f_j is known a priori as the evaluation $f(x_j)$. We take into account the limit case $j = 1$, and the same procedures can be applied for $j = n-2$:

$$\begin{aligned}(3.9) &\Rightarrow \frac{1}{h^2}(u_0 - 2u_1 + u_2) = f_1\\ (3.8) &\Rightarrow a - 2u_1 + u_2 = h^2 f_1\\ &\Rightarrow -2u_1 + u_2 = h^2 f_1 - a.\end{aligned}$$

The discrete problem can be expressed as an $(n-2)\times(n-2)$ matrix product without influencing the good properties of the matrix as:

$$AU = F$$

$$\Updownarrow$$

$$\begin{pmatrix} -2 & 1 & 0 & 0 & \ldots & 0 & 0 & 0 & 0\\ 1 & -2 & 1 & 0 & \ldots & 0 & 0 & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & 0 & 0 & \ldots & 0 & 1 & -2 & 1\\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1 & -2 \end{pmatrix} U = \begin{pmatrix} h^2 f_1 - a\\ h^2 f_2\\ \vdots\\ h^2 f_{n-3}\\ h^2 f_{n-2} - b \end{pmatrix}.$$

With U the interior unknowns vector described above.

Second approach

We present the approach used in this work where Dirichlet boundary conditions occurred. Even if this approach is simpler to apply, it does not ensure the retention of the matrix properties explained above; however, we will see that we handle particular matrices where is simpler to apply the conditions in this way, rather than disassemble the stencil on the right hand side for first interior points.

We consider the complete $n \times n$ matrix problem

$$\bar{A}\bar{U} = \bar{F}$$

$$\Updownarrow$$

$$\begin{pmatrix} 1 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0\\ 1 & -2 & 1 & 0 & \ldots & 0 & 0 & 0 & 0\\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & 0 & 0 & \ldots & 0 & 1 & -2 & 1\\ 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 1 \end{pmatrix} \bar{U} = \begin{pmatrix} a\\ h^2 f_1\\ \vdots\\ h^2 f_{n-2}\\ b \end{pmatrix},$$

where $\bar{U}$ is the complete set of unknowns. It is easy to prove that the two formulations are equivalent and that this second one corresponds to include the linear equations (3.8) in our system.

Neumann boundary conditions

Given the Neumann boundary conditions

$$u'(0) = a,\ u'(1) = b, \tag{3.10}$$

and considering, as above, the vectors of unknowns U and $\bar{U}$ for the discretized problem, we study the left boundary condition, remembering that the same reasoning can be applied on the right one, and for further dimensions, as done for the Dirichlet case.

First approach

A second-order way to approximate the Neumann boundary condition takes into account the existence of u_{-1}: a fictitious point that lies outside the interval such that the backward and forward difference formulation for the derivative $u'(0)$ can be expressed with $O(h)$ approximations:

$$u'(0) = \frac{u_1 - u_0}{h},\ u'(0) = \frac{u_0 - u_{-1}}{h}$$

$$\Downarrow$$

$$2u'(0) = \frac{u_1 - u_{-1}}{h}$$

$$\Downarrow (3.10)$$

$$\frac{1}{2h}(u_1 - u_{-1}) = a.$$

Then, we use the centred finite difference formula (3.9) to vanish this term not belonging on the discrete domain, obtaining a second-order accurate formulation

$$\frac{1}{h}(u_1 - u_0) = a + \frac{h}{2}f_0.$$

The obtained $n \times n$ matrix problem has form

$$\bar{A}\bar{U} = \bar{F}$$

$$\Updownarrow$$

$$\begin{pmatrix} -h & h & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & h & -h \end{pmatrix} \bar{U} = \begin{pmatrix} h(a + \frac{h}{2}f_0) \\ h^2 f_1 \\ \vdots \\ h^2 f_{n-2} \\ h(b + \frac{h}{2}f_{n-1}) \end{pmatrix}.$$

Second approach

The approach used in this work is first-order and follows the same principle using the approximations of boundary conditions (3.10) as first-order accurate formulations

$$u'(0) = \frac{u_1 - u_0}{h},$$
$$u'(1) = \frac{u_{n-1} - u_{n-2}}{h}.$$

The obtained $n \times n$ matrix problem has form

$$\begin{pmatrix} -1 & 1 & 0 & 0 & \dots & 0 & 0 & 0 & 0 \\ 1 & -2 & 1 & 0 & \dots & 0 & 0 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & \dots & 0 & 1 & -2 & 1 \\ 0 & 0 & 0 & 0 & \dots & 0 & 0 & -1 & 1 \end{pmatrix} \bar{U} = \begin{pmatrix} ha \\ h^2 f_1 \\ \vdots \\ h^2 f_{n-2} \\ hb \end{pmatrix}.$$

In mixed boundary condition cases, the approaches above can be easily combined in the matrix product formulation. This mixture is used in the current work applications.

3.1.3 Consistency, stability, and convergence

Let $L_h : \mathcal{M}_h \to D_h$ be a finite difference operator that discretizes a partial differential equation from a normed discrete sub-domain of $\mathbb{R}^n$ to a finite dimensional normed vector space:

$$L_h u_h = f_h \text{ in } \mathcal{M}_h. \tag{3.11}$$

u and u_h belong to different spaces, so we introduce the operator r_h such as $r_h u \in \mathcal{M}_h$. A natural choice, which we shall make, is that
$r_h : \mathcal{M} \subset \mathbb{R}^n \to \mathcal{M}_h$ is the orthogonal projection so that

$$||u - r_h u|| = \inf_{v \in \mathcal{M}_h} ||u - v||.$$

Definition 3.1.2 *A finite difference method is* ***convergent*** *when*

$$h \to 0 \Rightarrow ||r_h u - u_h||_{\mathcal{M}_h} \to 0.$$

Definition 3.1.3 *A finite difference method is* ***consistent*** *when*

$$h \to 0 \Rightarrow ||L_h(r_h u) - f_h||_{D_h} \to 0.$$

The consistency of a finite difference method does not imply the convergence in general [45]. In fact, the consistency represents a relationship between the numerical scheme and the differential equation. If the inverse operator L_h^{-1} is defined, we call the variable $c_h = ||L_h^{-1}||_{\mathcal{L}(\mathcal{M}_h, D_h)}$ *stability constant* of the discretization.

Definition 3.1.4 *A finite difference method is* ***stable*** *when* $\sup_h c_h < \infty$.

In other words, if h is the mesh width, the method is stable when $(L_h)^{-1}$ exists for all h sufficiently small ($h < h_0$) and if there is a constant C, independent of h, such that:

$$||(L_h)^{-1}|| \leq C \quad \forall h < h_0.$$

Theorem 3.1.3 *If a finite difference method is consistent and stable, then it is convergent.*

Taking as an example that the five-points stencil the consistency error is $O(h^2)$, the stability, and therefore convergence, is given by theorem 3.1.2. *Stability* means that the error caused by a small perturbation in the numerical solution remains constrained.

3.2 Finite Difference Methods on Hierarchical Grids

Although the principle of finite difference methods follows the criteria shown on a uniform mesh, the construction on hierarchical grids, where there are jumps of level, can follow different strategies. A centre-cell construction has been shown, but the first step when these methods are computed is to choose between nodes and cell-centred methods as needed.

Min et al. [16] chose a vertex-centred scheme, justified by their problem of a variable coefficient Poisson equation on non-graded octree-based grids. They computed the information on nodes, exploiting the presence of intermediate values computed on faces neighbours. The information on the intermediate values are computed using proper interpolations on the axial distances with fictitious points; then the finite difference method on the concerned node is calculated using a Taylor's analysis, where these interpolations are substituted, as mentioned above. As example, for the configuration in Figure 3.2, they involve the value on point u_4, thus obtained through a linear interpolation:

$$u_4 = \frac{s_5 u_6 + s_6 u_5}{s_5 + s_6}.$$

Taylor analysis gives the following results for the standard discretizations in the x and y directions, respectively:

$$(\frac{u_4 - u_0}{s_4} - \frac{u_0 - u_1}{s_1}) \cdot \frac{2}{s_1 + s_4} = u_{xx} + \frac{s_5 s_6}{(s_1 + s_4) \cdot s_4} u_{yy} + O(h)$$

$$(\frac{u_3 - u_0}{s_3} - \frac{u_0 - u_2}{s_2}) \cdot \frac{2}{s_2 + s_3} = u_{yy} + O(h)$$

With a linear weighting of spurious terms involving u_{yy} due to interpolations of the two equations above, the method is rendered second-order accurate. We invite the interested reader to [16] for a detailed analysis.

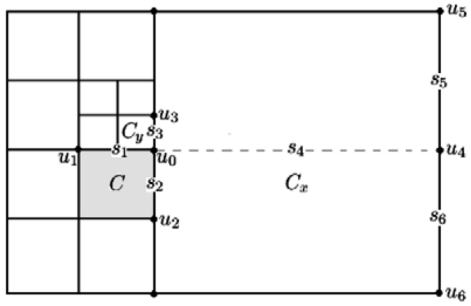


Figure 3.2: A two dimensional nodes centred discretization [16].

Berger and Colella [48] devised a cell-centred finite-difference scheme for shock hydrodynamics on an AMR mesh constituted by rectangular regions. The procedure follows an overlapping of refinement levels. The values on a grid at a given level are projected onto a virtual grid, coarsened by a factor of two in each direction with an error proportional to the truncation error on that point. Processing the discontinuities properly, they highlighted the memory management due to the AMR approach comparing the memory that would be used for a uniform mesh; in particular they present a factor of 2.2 larger than that required by AMR. Berger and Oliger [49] used a finite difference time integration for hyperbolic PDEs on rectangular subgrids with an adaptation of time step (larger or smaller) in line with the refinement of the grid. Moreover, their AMR approach is dynamic: every several time steps, they estimate the error at all grid points and adjust the grid structure. They also compared the time advantages with AMR, concluding that in less than at least one fourth of the time (in some cases), mesh refinement was able to calculate a solution that was accurate as the uniform fine grid calculation. For some tests, comparing also the complete procedure, they found that the entire run cost only 16% of the cost of a run on the uniform line grid with the same accuracy. Both these articles justify the choice of an AMR approach in terms of time and memory.

In [50] Oleg V. Vasilyev presents a high-order finite difference method on non-uniform meshes. His method preserves symmetries of the uniform mesh, and his conservation properties are as good as those of the standard second-order finite difference scheme on non-uniform meshes, while the accuracy is definitely superior. His formulation is valid on staggered grids and uses cell-centre values for the pressure points. It is not the only case where node and centre interpolations are mixed to compute several values; another example, fully conservative at second order for incompressible flow on non-uniform grids, is given by Ham et al. in [51].

Overall, a neighbour's search topology based on a cell-centre scheme shows more simple understanding. A scheme of this kind will be proposed in this thesis where, for level jumps, it has been chosen to respect geometry

without internal interpolations. This choice means that the symmetry of the uniform scheme can not be respected by making the resolution finite difference operator more complex. The difficulties of the operator, shown below, will be overwhelmed by the benefits of whole computation elements.

3.3 A Cell-Centred Finite Difference Method

In this section, we present the main idea to discretize the Laplacian. A similar approach is used for the gradients.

There are two natural choices to discretize differential operators on hierarchical grids: sampling at the nodes or at the centre of each cell. As an example, in [16], the discretization is vertex-centred, and in [27] it is cell centred. Here we considerer a cell-centred scheme. In this case, thanks to the data structure we use, the neighbour configuration is more easily accessible compared to a vertex-centred scheme.

The main idea is to ensure consistency and second-order accuracy of the truncation error in the sense of finite differences as a function of the number of neighbours. Let us focus for the moment on a two-dimensional problem and let us consider the configuration in Figure 3.3. As shown in [16], if only face-adjacent cells are to be used, then there is no locally consistent linear scheme in the sense of finite differences. Instead, we discretize the Laplace operator in c_4 using all the points belonging the first layer of neighbours. This will allow us to obtain more degrees of freedom than sufficient con-

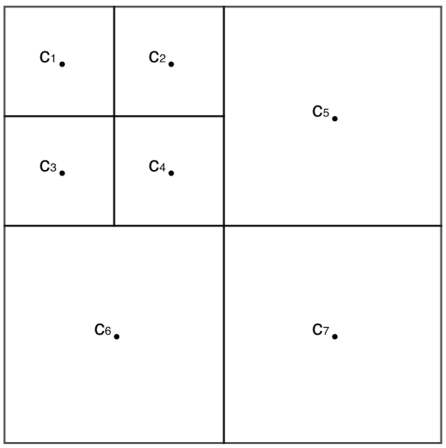


Figure 3.3: A test configuration centred in c_4. (Presented in [16])

straints for consistency. Possibly, as a function of the number of available points and symmetries, we will also ensure sufficient conditions for second-order accuracy. To see this, let h be the side length of the cell c_4. To obtain a consistent scheme, we must ensure that the discretization coefficients a_i,

$1 \leq i \leq 7$ satisfy:

$$u_{xx} + u_{yy} = a_1u_1 + a_2u_2 + a_3u_3 + a_4u_4 + a_5u_5 + a_6u_6 + a_7u_7 + O(h).$$

Using standard Taylor analysis, we expand the solution u_i in c_i with respect to u_4 in c_4, $\forall i = 1, \ldots, 7$, and get:

$$u_4 = u_4,$$

$$u_1 = u_4 - h\frac{\partial u_4}{\partial x} + h\frac{\partial u_4}{\partial y} + \frac{h^2}{2}\frac{\partial^2 u_4}{\partial x^2} - h^2\frac{\partial^2 u_4}{\partial x \partial y} + \frac{h^2}{2}\frac{\partial^2 u_4}{\partial y^2} + O(h^3),$$

$$u_2 = u_4 - h\frac{\partial u_4}{\partial y} + \frac{h^2}{2}\frac{\partial^2 u_4}{\partial y^2} + O(h^3),$$

$$u_3 = u_4 - h\frac{\partial u_4}{\partial x} + \frac{h^2}{2}\frac{\partial^2 u_4}{\partial x^2} + O(h^3),$$

$$u_5 = u_4 + \frac{3h}{2}\frac{\partial u_4}{\partial x} + \frac{h}{2}\frac{\partial u_4}{\partial y} + \frac{9h^2}{8}\frac{\partial^2 u_4}{\partial x^2} + \frac{3h^2}{4}\frac{\partial^2 u_4}{\partial x \partial y} + \frac{h^2}{8}\frac{\partial^2 u_4}{\partial y^2} + O(h^3),$$

$$u_6 = u_4 - \frac{h}{2}\frac{\partial u_4}{\partial x} - \frac{3h}{2}\frac{\partial u_4}{\partial y} + \frac{h^2}{8}\frac{\partial^2 u_4}{\partial x^2} + \frac{3h^2}{4}\frac{\partial^2 u_4}{\partial x \partial y} + \frac{9h^2}{8}\frac{\partial^2 u_4}{\partial y^2} + O(h^3),$$

$$u_7 = u_4 + \frac{3h}{2}\frac{\partial u_4}{\partial x} - \frac{3h}{2}\frac{\partial u_4}{\partial y} + \frac{9h^2}{8}\frac{\partial^2 u_4}{\partial x^2} - \frac{9h^2}{4}\frac{\partial^2 u_4}{\partial x \partial y} + \frac{9h^2}{8}\frac{\partial^2 u_4}{\partial y^2} + O(h^3).$$

A complete Taylor analysis, using the coefficients of the Taylor expansions as columns of a matrix for all the involved neighbours, leads to the following linear system:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & -h & 0 & -h & \frac{3h}{2} & -\frac{h}{2} & \frac{3h}{2} \\ 0 & h & h & 0 & \frac{h}{2} & -\frac{3h}{2} & -\frac{3h}{2} \\ 0 & \frac{h^2}{2} & 0 & \frac{h^2}{2} & \frac{9h^2}{8} & \frac{h^2}{8} & \frac{9h^2}{8} \\ 0 & -h^2 & 0 & 0 & \frac{3h^2}{4} & \frac{3h^2}{4} & -\frac{9h^2}{4} \\ 0 & \frac{h^2}{2} & \frac{h^2}{2} & 0 & \frac{h^2}{8} & \frac{9h^2}{8} & \frac{9h^2}{8} \end{pmatrix} \begin{pmatrix} a_4 \\ a_1 \\ a_2 \\ a_3 \\ a_5 \\ a_6 \\ a_7 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

In the example above we must determine seven discretization coefficients a_i, $1 \leq i \leq 7$, but we only have six constraints for consistency. The idea is to ensure consistency and, at the same time, to minimize the deviation from second-order accuracy as follows.

In the general case, when the number of constraints is m, we solve the constrained minimization problem by defining an appropriate Lagrangian function. Let $\lambda \in \mathbb{R}^m$ a vector of Lagrange multipliers, $a \in \mathbb{R}^n$ the discretization coefficient vector of size n (the size of the stencil), $M \in \mathcal{M}^{m,n}(\mathbb{R})$ the constraint matrix, $f \in \mathbb{R}^m$ the right hand side vector corresponding to

the imposed constraints and $F(a)$ a convex cost function from $\mathbb{R}^n$ to $\mathbb{R}$. We define a Lagrangian function $\mathcal{L}(a,\lambda) : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ as follows

$$\mathcal{L}(a,\lambda) = F(a) - \lambda^T(Ma - f), \tag{3.12}$$

and compute the stationary point of this function with respect to (a,λ):

$$\begin{cases} \frac{\partial \mathcal{L}(a,\lambda)}{\partial a} = & 0 \\ \frac{\partial \mathcal{L}(a,\lambda)}{\partial \lambda} = & 0 \end{cases} \Leftrightarrow \begin{cases} \frac{\partial F}{\partial a} - M^T\lambda & = 0 \\ Ma & = f. \end{cases}$$

Let $B \in \mathcal{M}^{6,n}$ be the submatrix corresponding to the consistency constraints, $C \in \mathcal{M}^{4,n}$ be the submatrix relative to the second-order constraints and $\alpha \in [0,1]$. The matrix C is obtained adding terms to the Taylor's expansion until third order, as done for the consistency constraints, taking into account the configuration in Fig.3.3 is:

$$\begin{pmatrix} 0 & -\frac{h^3}{6} & 0 & -\frac{h^3}{6} & \frac{27h^3}{48} & -\frac{h^3}{48} & \frac{27h^3}{48} \\ 0 & \frac{h^3}{2} & 0 & 0 & \frac{9h^3}{16} & -\frac{3h^3}{16} & -\frac{27h^3}{16} \\ 0 & \frac{h^3}{2} & 0 & 0 & \frac{3h^3}{16} & -\frac{9h^3}{16} & -\frac{27h^3}{16} \\ 0 & \frac{h^3}{6} & \frac{h^3}{6} & 0 & \frac{h^3}{48} & -\frac{27h^3}{48} & -\frac{27h^3}{48} \end{pmatrix}$$

The discretization coefficients are then rescaled, dividing by the appropriate value of the cell side. We distinguish two cases:

- $n \leq 10$: $M = B$ and we take $F(a) = 1/2a^T\left((1-\alpha)C^TC + \alpha I\right)a$ and the local system to be solved is

$$\begin{pmatrix} ((1-\alpha)C^TC + \alpha I) & -B^T \\ B & 0 \end{pmatrix}\begin{pmatrix} a \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ f \end{pmatrix}.$$

  This choice of the convex function $F(a)$ is such that the discretization coefficients minimize the second-order truncation error encoded in matrix C, and their L_∞ norm is penalized by coefficient α. We have chosen a small value of α that results in a stable matrix to invert and that introduces the minimal amount of regularisation. We took $\alpha = 0.01$ for all the numerical illustrations in the following The coefficients a always satisfy 6 consistency constraints.

- $n > 10$: $M = \binom{B}{C}$ and we take $F(a) = 1/2a^T a$, $m = 10$, $I \in \mathcal{M}_{n,n}$, and the local system to be solved is

$$\begin{pmatrix} I & -M^T \\ M & 0 \end{pmatrix}\begin{pmatrix} a \\ \lambda \end{pmatrix} = \begin{pmatrix} 0 \\ f \end{pmatrix}.$$

  The coefficients satisfy 10 second-order accuracy constraints while their L_∞ norm is minimized.

This approach is independent of the specific grid configuration and can be applied to either graded or non-graded grids. Although we use a cell-centred stencil, this method can in principle be applied to vertex-centred stencils. We remark that the minimal number of available points, including only the first neighbours in 2D, is 7 if the grid is graded and 6 if the grid is non-graded. Therefore, the discretization will always be at least consistent.

3.3.1 Three-dimensional extension

The neighbours are found through faces, edges and vertexes. The consistency constraints are 10, the number of equations to obtain second order accuracy is 20. For either graded or non-graded grids, the scheme will be at least consistent since with graded grids we have at least 15 available points, including only first layer neighbours, and 11 with non-graded cases. Beyond 30 available points, in order to limit the size of the stencil, we consider the minimum number of all possible neighbours satisfying consistency and second-order accuracy.

3.3.2 Speed up the computation

On graded grids, our tool for identifying the configurations, introduced in chapter 2, allows us a gain of memory and time for all internal communications that requires knowing the structure of the neighbourhood but not necessarily the position or dimension. In fact, the strategy to build the minimization problem is handled from an arbitrary value of h and then adapted to perform the matrix condition number as best as possible when the depth of the tree becomes relevant; also the operations to recognize a uniform neighbourhood through faces, and some others, are independent from the topological position of the configuration in space and processes. These observations can now explain the utility to have a bijection between the configuration and integer array.

3.3.3 Remarks on the uniform stencil

In 3.1.1, we described the classical five points stencil with its regular matrix, then in 3.1.3 we presented some important properties of finite difference schemes. As explained, our Morton code won't let us obtain a matrix so regular; meanwhile we can ensure the local consistency, but in general, the stability proof of a finite difference scheme requires a complex Fourier analysis [52]. For time-dependant PDEs, some conditions to ensure stability can be applied. Arnold [53], Lax and Richtmyer [54] studied the problem in detail; however, as far as we know, there are not local properties that can ensure the stability for our method.

We propose to solve a typical uniform zone using all neighbours involved. The linear system built following our method assumes the form:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & -h & h & 0 & 0 & -h & h & -h & h \\ 0 & 0 & 0 & -h & h & -h & -h & h & h \\ 0 & \frac{h^2}{2} & \frac{h^2}{2} & 0 & 0 & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} \\ 0 & 0 & 0 & 0 & 0 & h^2 & -h^2 & -h^2 & h^2 \\ 0 & 0 & 0 & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}. \tag{3.13}$$

This problem has ∞^3 possible solutions in the set of:

$$\{a_0 = -\frac{4}{h^2} + 2a_6 + 2a_7,\ a_1 = \frac{1}{h^2} - a_5 - a_7,\ a_2 = \frac{1}{h^2} + a_5 - 2a_6 - a_7,$$
$$a_3 = \frac{1}{h^2} - a_5 - a_6,\ a_4 = \frac{1}{h^2} + a_5 - a_6 - 2a_7,\ a_8 = -a_5 + a_6 + a_7\}.$$

We can then try to determine a second-order solution of problem (3.13), trying to solve:

$$\begin{pmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & -h & h & 0 & 0 & -h & h & -h & h \\ 0 & 0 & 0 & -h & h & -h & -h & h & h \\ 0 & \frac{h^2}{2} & \frac{h^2}{2} & 0 & 0 & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} \\ 0 & 0 & 0 & 0 & 0 & h^2 & -h^2 & -h^2 & h^2 \\ 0 & 0 & 0 & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} & \frac{h^2}{2} \\ 0 & -\frac{h^3}{6} & \frac{h^3}{6} & 0 & 0 & -\frac{h^3}{6} & \frac{h^3}{6} & -\frac{h^3}{6} & \frac{h^3}{6} \\ 0 & 0 & 0 & 0 & 0 & -\frac{h^3}{2} & -\frac{h^3}{2} & \frac{h^3}{2} & \frac{h^3}{2} \\ 0 & 0 & 0 & 0 & 0 & -\frac{h^3}{2} & \frac{h^3}{2} & -\frac{h^3}{2} & \frac{h^3}{2} \\ 0 & 0 & 0 & -\frac{h^3}{6} & \frac{h^3}{6} & -\frac{h^3}{6} & -\frac{h^3}{6} & \frac{h^3}{6} & \frac{h^3}{6} \end{pmatrix} \begin{pmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \\ a_6 \\ a_7 \\ a_8 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}. \tag{3.14}$$

The set of ∞^1 solutions for (3.14) is:

$$\{a_0 = 4a_5 - \frac{4}{h^2},\ a_1 = \frac{1}{h^2} - 2a_5, a_2 = \frac{1}{h^2} - 2a_5,\ a_3 = \frac{1}{h^2} - 2a_5,\ a_4 = \frac{1}{h^2} - 2a_5,$$
$$a_6 = a_5, a_7 = a_5,\ a_8 = a_5\}.$$

The discretization weights (Fig. 3.4) obtained with our minimization problem are:

- $\frac{-1.\bar{3}}{h^2}$ for the red point (centre of the configuration);
- $\frac{-0.\bar{3}}{h^2}$ for the face adjacent cells marked in green;

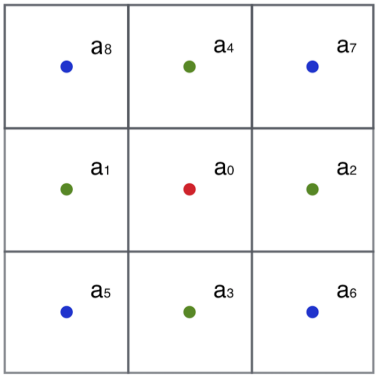


Figure 3.4: Uniform mesh configuration. The weights are enumerated following the *internal* Z-Order of a_0 through faces, then through vertices.

- $\frac{0.\bar{6}}{h^2}$ for the corner blue points.

The resulting truncation error, easily obtained substituting the weights in (3.14), ensures the order two of convergence; however, this nine-point stencil is an example of a second-order scheme, but it is possibly not stable. As long as we can not prove the stability, our choice falls on the five-points stencil, allowing us a gain of elements in the Laplacian operator. We do not report here some results about it, but we specify that nine-points stencils are useful to obtain high-order approximations precisely because of the flexibility in the infinite set of solutions [46]. The additional constraints can be, in fact, adapted to such specific complex problems as heat conduction, Helmhotz equations, and so on (see, for example, [55], [56], [57]). The observations in this section are also valid for the three-dimensional case.

3.4 Preliminary Conclusions

We explained quadtree data structures' application with some different discretizations that could be applied on these kinds of meshes (chapter 2). In this part, we explored the existent finite-difference methods conceived similarly, or on similar meshes, if compared with ours.

Most finite volume methods are cell centred, due to fluxes node-sampling gradients; meanwhile, for finite-difference method, it is the opposite. An analysis of existent methods is proposed by C. Batty [[58], p.5] who introduces a finite-volume method on non-graded grids. Our method can be compared with the others, not only for its local consistency (with tendency towards the second order of convergence globally) but also for the ease with which it manages the structure along jumps.

The method proposed in the current work is among the first finite dif-

ference schemes to devise the weights stencils without the use of internal interpolations and fictitious points.

Chapter 4

Numerical Results

In this chapter, the numerical results are presented. We start with a proof of consistency to validate our method in two and three dimensions. We also analyse ill-posed configurations in other methods to demonstrate the advantages of our approach. In three dimensions, an analysis of convergence of the sparse operator derived by the discretization is given. A comparison with other finite volume methods is briefly proposed in two dimensions, concluding with unbalanced mesh cases to examine the sTable computation of weights if the geometry is abruptly changing.

We will discuss the discontinuity approach, first of all with penalization tests, to simulate the internal boundary conditions, and we will present the strategy on various cases in two and three dimensions. Moreover, we provide a comparison of time and memory with an uniform regular solver to highlight that for equal error there is better performance and an evident memory advantage. Then, we perform a study on internal discontinuities; we treat them as mollified functions, continuous between two values, and we expose a first simple approach for internal jumping conditions.

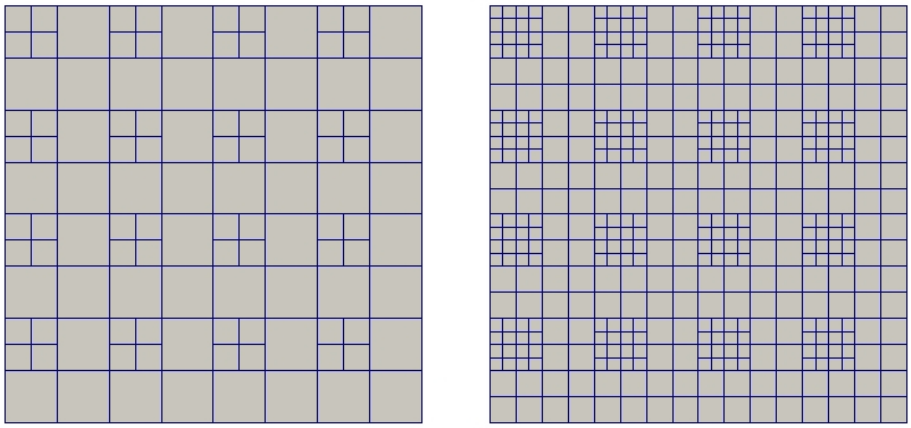
The final part of this chapter contains details about the code structure: the handling of the parallelization, and the solvers used, concluding with its properties of scalability.

4.1 Numerical Results: Consistency

4.1.1 Two-Dimensional Test

We consider here the case $\Delta u = f$. The domain is the two-dimensional square, $\Omega = [0,1] \times [0,1]$, and the grid studied is a biperiodic lattice obtained by initially repeating the elementary configuration presented in Chapter 3, Fig. 3.3, obtaining the mesh in Figure 4.1.We have seen that for this grid, a scheme that does not include all the first neighbours' layer, but only face-adjacent cells, is always inconsistent, as shown in [16].

The analytical solution considered is $u_e(x_1, x_2) = \sin\left((x_1 - 0.5)^2 + (x_2 - 0.5)^2\right)$, and the problem solved is $\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}) = \Delta u_e(\boldsymbol{x})$, with exact Dirichlet boundary conditions imposed at the cell centres along the border of the domain. The convergence rate is obtained by subsequently subdividing each cell, and it is given in Table 4.1. Second-order accuracy is obtained as anticipated. An example of the error distribution over a refined grid is presented in Figure 4.2.



(a) Tree level 4 (b) Tree level 5

Figure 4.1: Example of initial grid and its subsequent refinement.

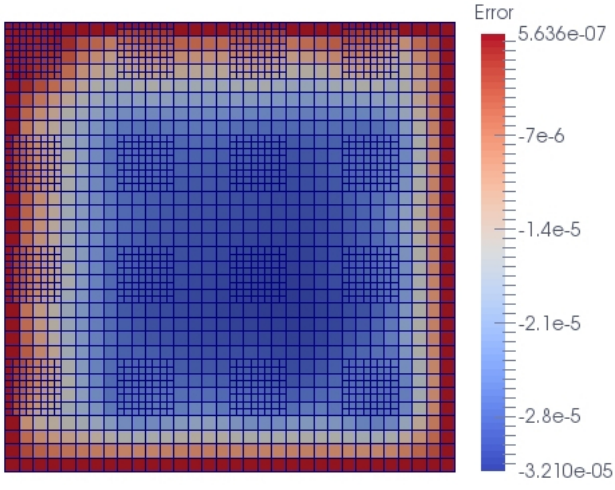


Figure 4.2: Example of error distribution on a grid corresponding to tree level 6.

A second proof of convergence, regarding the same test configuration,

Tree level	L^∞	Order	L^2	Order
4	$5.023 \cdot 10^{-4}$		$2.491 \cdot 10^{-4}$	
5	$9.921 \cdot 10^{-5}$	2.5315	$4.97 \cdot 10^{-5}$	2.506
6	$2.239 \cdot 10^{-5}$	2.2155	$1.131 \cdot 10^{-5}$	2.19725
7	$5.364 \cdot 10^{-6}$	2.087	$2.725 \cdot 10^{-6}$	2.075
8	$1.317 \cdot 10^{-6}$	2.0365	$6.709 \cdot 10^{-7}$	2.0305

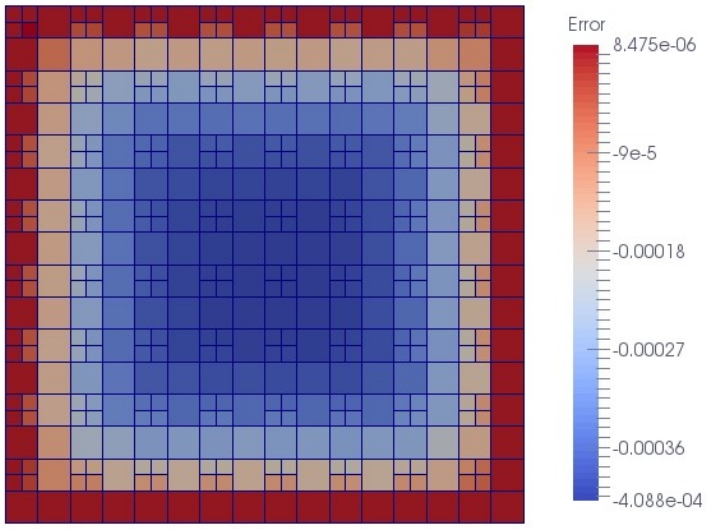
Table 4.1: Error norms and order of the scheme. Proof of consistency referring to Fig. 4.1, 4.2

consists of the repetition of the test configuration, increasing the depth of our tree (Fig. 4.3). The order of convergence for this case is given in Table 4.2.

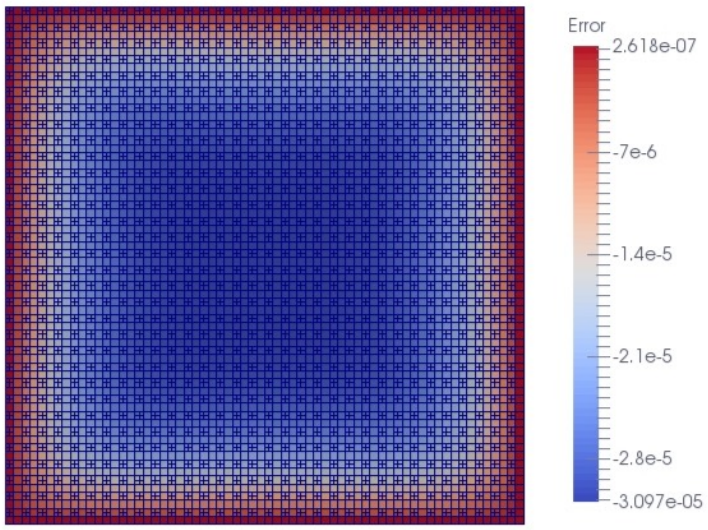
Tree level - Octants	L^∞	L^2	Order
5 - 448	$4.088 \cdot 10^{-4}$	$2.679 \cdot 10^{-4}$	-
6 - 1792	$1.160 \cdot 10^{-4}$	$8.038 \cdot 10^{-5}$	1.737
7 - 7168	$3.097 \cdot 10^{-5}$	$2.195 \cdot 10^{-5}$	1.873
8 - 28672	$7.943 \cdot 10^{-6}$	$5.698 \cdot 10^{-6}$	1.946
9 - 114688	$2.012 \cdot 10^{-6}$	$1.448 \cdot 10^{-6}$	1.976
10 - 458752	$5.176 \cdot 10^{-7}$	$3.714 \cdot 10^{-7}$	1.963

Table 4.2: Error norms and order of the scheme. Proof of consistency referring to Fig. 4.3.

The two cases studied allow us to demonstrate the stability of the method and, above all, the consistency. The configuration analysed was chosen for its property to not be consistent if only face-adjacent cells are used (as seen in Section 3.3, [16]). In the first refinement (Fig. 4.2) we can see the second-order convergence; the tendency of second order accuracy is also presented for the second refinement approach (Fig. 4.3). The reason for this inequality is the occurrence of the uniform stencil (second-order truncation error) combined with the occurrence of the configuration presented, where the consistency is ensured and the second order is minimized. We can conclude that this method is always consistent, with a global tendency to the second order.



(a) Tree level 5



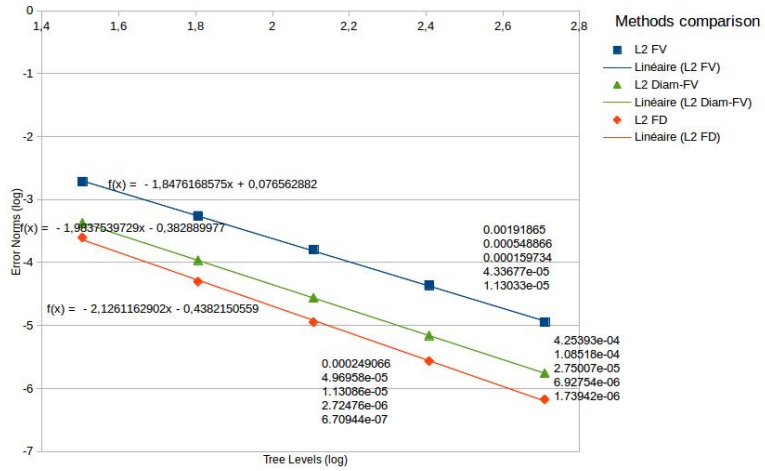
(b) Tree level 7

Figure 4.3: Examples of error distribution on a grid corresponding to two different levels. Simple repeat test configuration for each level of refinement.

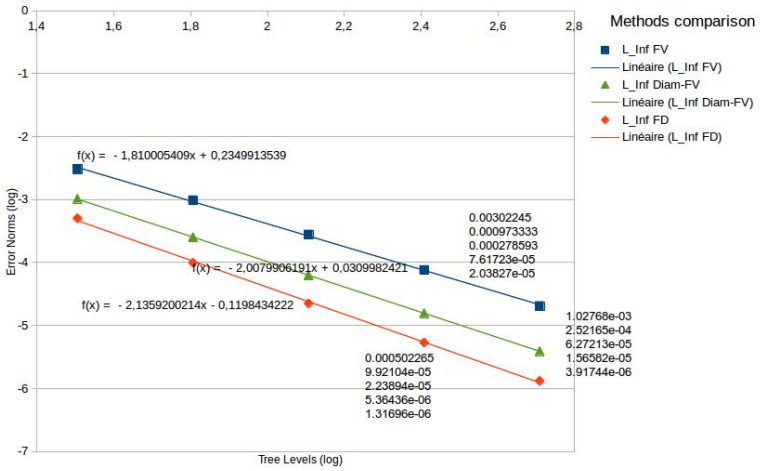
Comparison With Other Methods

The finite different method presented in this work has been compared with two other finite volume schemes implemented by *Memphis Inria* project-team members. The results of this test are given in Figure 4.4). The first method, identified by *FV*, was implemented by Marco Cisternino [59]; the second one, identified by *Diam FV*, by Claire Taymans (Morel). All three

methods stick on the same grid (Fig. 4.2), and they use PABLO's data structure with its parallel balance.



(a) L2 norm comparison



(b) LInf norm comparison

Figure 4.4: Norms Comparison

The finite volume scheme to compute gradients on the intersections hybridises the method averaging cell gradients of neighbours and introduces a finite difference correction along the direction joining neighbour cell centres. In the conforming region, the truncation error of the discrete elliptic operator is $O(h2)$ because it is the classical second-order centred scheme. On the other hand, the truncation error is only $O(1)$ in the non-conforming region because the fluxes are approximated with only first-order accuracy and there is not cancellation effect due to symmetry, allowing to preserve the first-order accuracy for the elliptic operator, as is the case in the conforming

region.

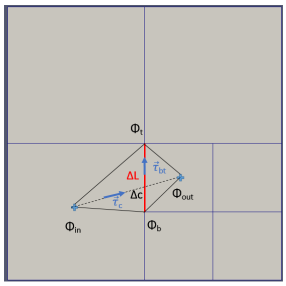


Figure 4.5: Diamond' scheme stencil

The second method of this comparison uses the diamond scheme, when a gap in refinement level occurs between two neighbours. It consists of defining a dual mesh (Fig. 4.5) and assumes that the gradient is constant inside the diamond. Solving the equations system (4.1),(4.2), it returns the gradients values through the cells faces.

$$\begin{cases} \boldsymbol{\nabla}\Phi.\boldsymbol{\tau_c} = \dfrac{\Phi_{out} - \Phi_{in}}{\Delta_c} \\ \boldsymbol{\nabla}\Phi.\boldsymbol{\tau_{bt}} = \dfrac{\Phi_t - \Phi_b}{\Delta_L} \end{cases} \tag{4.1}$$

$$\begin{cases} \dfrac{\partial\Phi}{\partial x}.\tau_{cx} + \dfrac{\partial\Phi}{\partial y}.\tau_{cy} = \dfrac{\Phi_{out} - \Phi_{in}}{\Delta_c} \\ \dfrac{\partial\Phi}{\partial x}.\tau_{btx} + \dfrac{\partial\Phi}{\partial y}.\tau_{bty} = \dfrac{\Phi_t - \Phi_b}{\Delta_L}. \end{cases} \tag{4.2}$$

This comparison points out that all three methods applied to the same problem are in the second order of convergence, with somewhat different local precision. Indeed, the finite difference method presented in this work, as it optimises the truncation error locally depending on the neighbourhood configuration, tends to be more accurate than the finite volume methods that look at the problem globally.

Unbalanced Cases

We conclude the two-dimensional results for this part by presenting some of unbalanced tests.

In the first case presented (Fig. 4.6, Table 4.3) we split the domain in two equal parts and impose a level jump between the subdomains of 4 without balancing. In this test we solved the equation:

$$\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}) \quad \boldsymbol{x} \in \Omega, \tag{4.3}$$

imposing Dirichlet's boundary condition. This test allows us to exploit the weights calculation when strong internal jumps are present.

The analytical solution we used to study the convergence is the paraboloid: $u_e(\boldsymbol{x}) = u_e(x_1, x_2) = \frac{1}{8} - \frac{1}{4}((x_1 - 0.5)^2 + (x_2 - 0.5)^2)$.

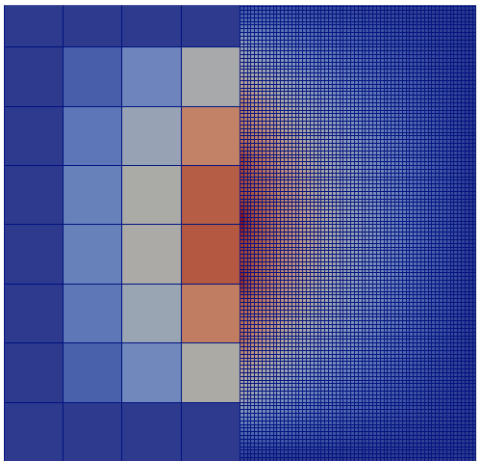


Figure 4.6: Error distribution example. Unbalanced case mesh.

Norm Infty	Norm 2	Mesh Points	Solver Iterations
$7.500 \cdot 10^{-8}$	$1.611 \cdot 10^{-7}$	8224	154
$2.453 \cdot 10^{-8}$	$7.679 \cdot 10^{-8}$	32896	333
$3.220 \cdot 10^{-8}$	$1.20 \cdot 10^{-7}$	131584	695
$4.361 \cdot 10^{-8}$	$2.187 \cdot 10^{-7}$	526336	8995
$4.899 \cdot 10^{-8}$	$2.486 \cdot 10^{-7}$	2105344	107364
$5.095 \cdot 10^{-8}$	$2.594 \cdot 10^{-7}$	8421376	734596

Table 4.3: Error norms. Case in Fig.4.6

We can remark, in Table 4.3, that if we do not force the tolerances of the solver, the error becomes stable without changing solver tolerances, for this solution, so it is impossible to see an order of convergence. We used a block Jacobi preconditioning (BJACOBI) on a global flexible GMRES. For this purpose, we tested a different mesh with a circular sub-domain of discontinuity, without the imposition of balance constraints (Fig. 4.7). In Table 4.4, the error variation is more evident for a sinusoidal analytical solution: $u_e(x_1, x_2) = \sin\left((x_1 - 0.5)^2 + (x_2 - 0.5)^2\right)$.

These tests conclude for the two-dimensional part the proof of the consistency of the method. For topological and computational reasons (that will be presented below), if not required, it is preferred to balance the structure, although it can be seen from the comparison between Tables 4.2 and 4.4 the robustness of the convergence order.

Remark 4.1.1 (On the order calculations) *Convergence orders in AMR cases, especially where we try to track an interface, are approximated fol-*

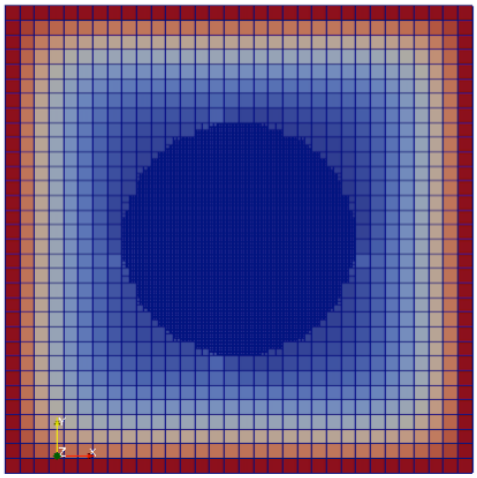


Figure 4.7: Error distribution example, level 8, sinus analytical function.

Norm Infty	Norm 2	Mesh Points	Order	Solver Iterations
$1.388 \cdot 10^{-3}$	$6.825 \cdot 10^{-3}$	796	-	123
$9.973 \cdot 10^{-5}$	$5.503 \cdot 10^{-4}$	3352	1.752	167
$2.738 \cdot 10^{-5}$	$1.596 \cdot 10^{-4}$	13588	1.769	352
$7.789 \cdot 10^{-6}$	$4.559 \cdot 10^{-5}$	54268	1.809	727
$1.955 \cdot 10^{-6}$	$1.141 \cdot 10^{-5}$	217936	1.994	161905

Table 4.4: Error norms and order. Case in Fig.4.7

lowing a dimensional-dependent formulation: $p = d * \frac{\ln(err_1/err_2)}{\ln(np_2/np_1)}$, *where* d *stands for dimension,* err_i *for error norms and* np_i *is the total number of concerned points. We should remark, in general, that two meshes subsequent in level of refinement may not be exactly the same; the number of points is not a simple increase for four (eight in 3D, as example see last two cases of Table 4.4) but it is subject to small variations that still give a reliable formula.*

4.1.2 Three Dimensional Test

As explained in Chapter 3-Remark 3.3.1, the computational details aimed at building the finite difference method for the three-dimensional extension follow the same strategy. The number of constraints necessary for a consistent set of weights becomes 10 instead of 6; similarly, the dimensions of the local variables have to be increased properly. We show in this section the convergence and consistency of the three-dimensional method so as to complete the proof.

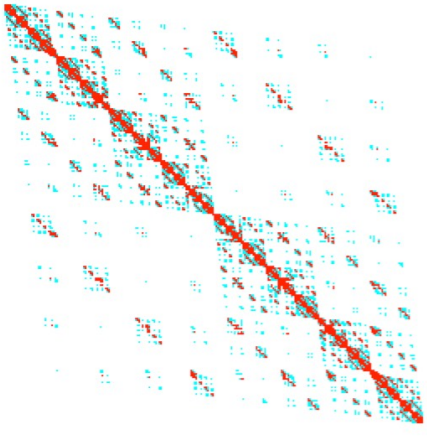


Figure 4.8: 3D matrix draw. 36128 points, tree's level 6.

In Figure 4.8 we present an example of non-zero elements entries in the solver matrix for an AMR which follows a centred sphere with one level of jump between the two parts involved (Fig.4.9); we can highlight the sparsity of the matrix that presents several sub-blocks of non-zeros. This sparsity is due to the Z-order, a problem repeatedly faced in previous sections.

As in previous cases, we solve the problem $\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega$ imposing Dirichlet's boundary conditions on $\Omega = [0,1] \times [0,1] \times [0,1]$, at first, using an analytical solution invariant along the third axis $u_e(x_1, x_2, x_3) = \sin\left((x_1 - 0.5)^2 + (x_2 - 0.5)^2\right)$. We set an internal spherical sub-domain where the mesh assumes one level more than elsewhere. In Figure 4.9, a section of the error for this case is presented. Confirming our previous two-dimensional studies, the error is concentrated in the jumping zones between the different levels of the mesh.

We intend to study the convergence of our operator. Let Δ_h be the Laplacian operator of our problem and u_e the analytical solution calculated on the cells' centres. We present in Table 4.5 an analysis of the residual of the operator $\Delta_h \cdot u_e - f_e$, and a section of it cell by cell is showed in Figure 4.10. Once we study the convergence of the operator, we emulate the cases proposed in the two-dimensional part by increasing the level of jumps on a

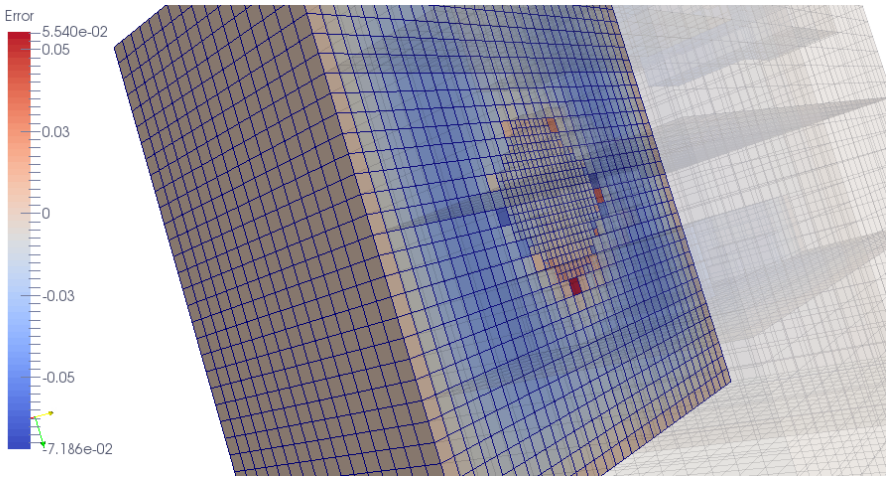


Figure 4.9: Distribution of error on a AMR following a central sphere with radius 0.15.

Tree's depth level	$\|\|\Delta_h \cdot u_a - f\|\|_\infty$	Mesh Points	Order
5	$7.325{\cdot}10^{-3}$	4488	
6	$2.207{\cdot}10^{-3}$	36128	1.726
7	$5.981{\cdot}10^{-4}$	287680	1.888
8	$1.552{\cdot}10^{-4}$	2303400	1.945

Table 4.5: Study of residual order.

balanced mesh from one to three. We always solve the equation $\Delta u(\boldsymbol{x}) = f(\boldsymbol{x})$ with Dirichlet's boundary conditions; the results are presented in Table 4.6.

Norm Infty	Norm 2	Mesh Points	Order
$2.412{\cdot}10^{-4}$	$1.162{\cdot}10^{-4}$	4880	
$7.936{\cdot}10^{-5}$	$4.161{\cdot}10^{-5}$	32656	1.6184
$2.201{\cdot}10^{-5}$	$1.235{\cdot}10^{-5}$	264944	1.7461
$5.829{\cdot}10^{-6}$	$3.360{\cdot}10^{-6}$	2111880	1.8813
$1.501{\cdot}10^{-6}$	$8.761{\cdot}10^{-7}$	17103976	1.9281
$3.807{\cdot}10^{-7}$	$2.236{\cdot}10^{-7}$	137484026	1.965

Table 4.6: Laplacian resolution AMR in a sphere. Balanced mesh, three levels of difference between maximal and minimal depth; two-dimensional sinus function.

In Figure 4.11 we extract the two different parts for a level of refinement 9 in the sphere and 6 outside. At this level of refinement we can highlight the symmetry of the error in the uniform areas. The radius of the sphere is

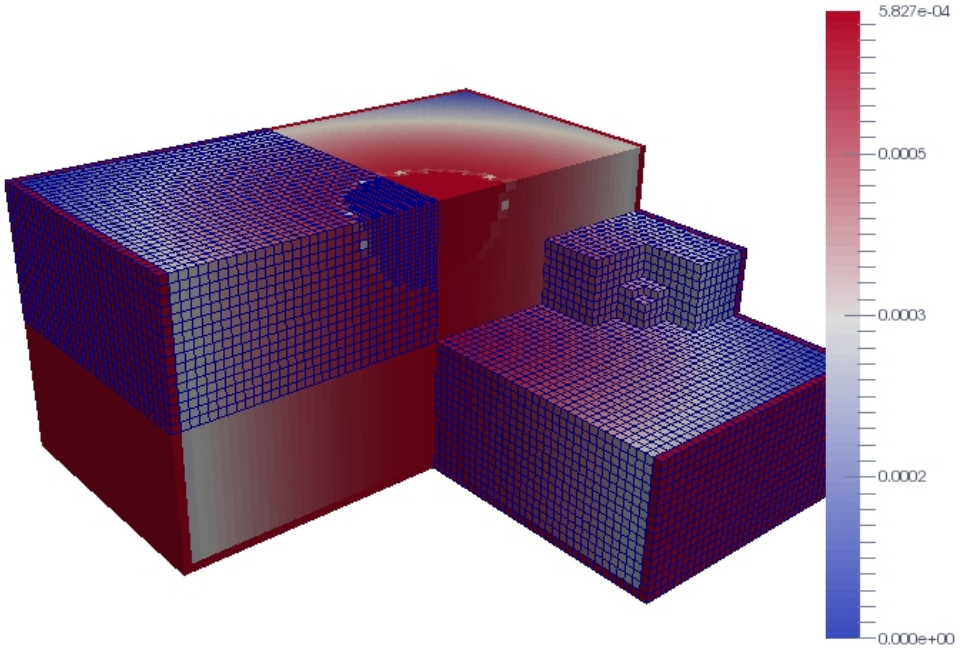


Figure 4.10: Section of the domain studying the residual.

0.15.

We remind that for this test, the analytical solution is bi-dimensional; this case allowed us to determine the stability of the third axial contribution of weights with the new existence of edge directions; as shown, they don't influence the computation until they are not taken into account. To conclude the test of consistency, we need a three-dimensional evaluation. We consider $\Delta u(\boldsymbol{x}) = f(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega$ imposing Dirichlet's boundary conditions on $\Omega = [0,1] \times [0,1] \times [0,1]$, and we apply the analytical solution $u_e(x_1, x_2, x_3) = \sin\left((x_1 - 0.5)^2 + (x_2 - 0.5)^2 + (x_3 - 0.5)^2\right)$. As above, we present in Table 4.7 the convergence rate, and in Figure 4.12 the error outside and inside the considered sphere.

Norm Infty	Norm 2	Mesh Points	Order
$3.937{\cdot}10^{-4}$	$1.977{\cdot}10^{-4}$	4880	
$1.257{\cdot}10^{-4}$	$7.147{\cdot}10^{-5}$	32656	1.61
$3.548{\cdot}10^{-5}$	$2.121{\cdot}10^{-5}$	264944	1.74
$9.426{\cdot}10^{-6}$	$5.777{\cdot}10^{-6}$	2111880	1.878
$2.428{\cdot}10^{-6}$	$1.506{\cdot}10^{-6}$	17103976	1.935

Table 4.7: Errors: Laplacian resolution AMR in a sphere. Balanced mesh, three levels of difference between maximal and minimal depth; three-dimensional sinus function.

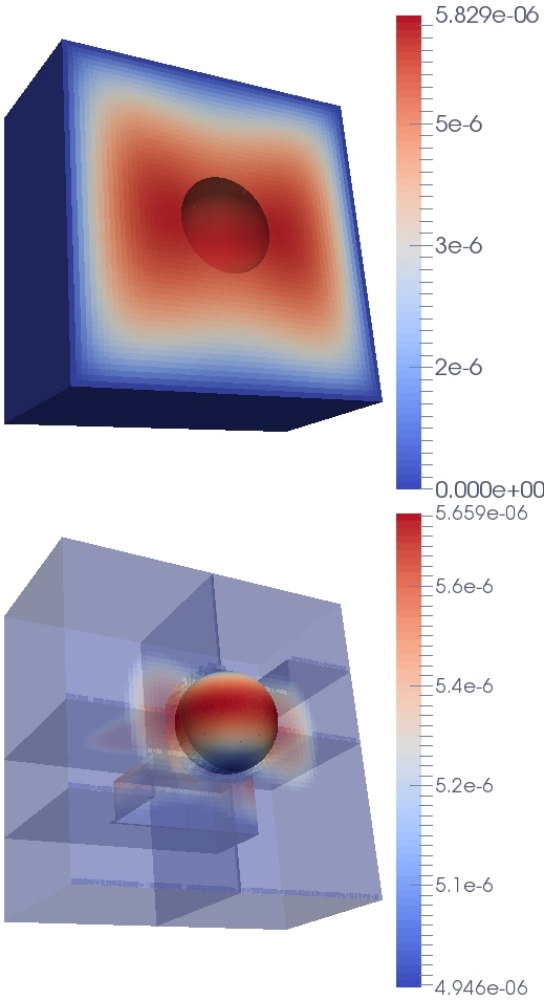


Figure 4.11: Distribution of error around and inside the sphere. Tree's levels 9 inside and 6 outside.

These results conclude the demonstration part on the finite difference method. To test the gradients approximations, similar tests were repeated for the respective derivatives. We can conclude beforehand that the method, both in two and three dimensions, is consistent with the second-order trend.

4.1.3 The Mesh Refinement

The previous examples, and most of those we see, follow the same refinement strategy around a subdomain. Unless specified otherwise, there will be three level jumps between the maximum and minimum depth of the tree. Given $\phi(\boldsymbol{x})$ the distance function with the considered sub-domain boundary, and a tolerance δ sufficiently small, the mesh refinement is defined as:

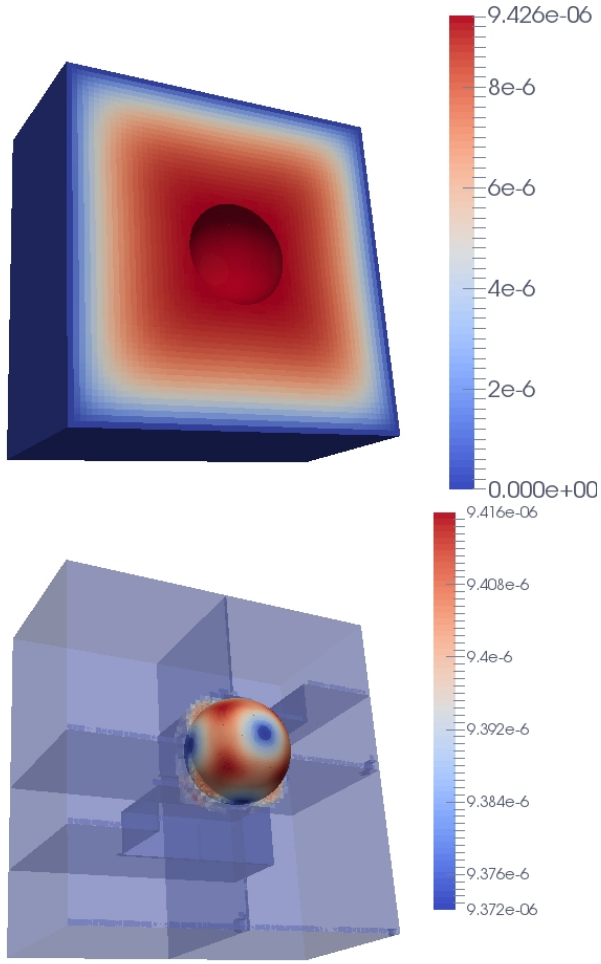


Figure 4.12: Distribution of error around and inside the sphere. Tree's levels 9 inside and 6 outside. Three-dimensional solution.

- the maximal depth of the tree is fixed at value M;
- the squared domain is uniformly refined until level $M - 3$;
- from $M - 3$ to M the incidence with the internal sub-domain is evaluated on each octant. If the octant lies on the refining zone, that means $|\phi(x)| < \delta$, and it splits in four/eight children.
- balance constraints are applied on the jump zones.

4.2 Dirichlet boundary conditions and penalization

Internal boundaries with Dirichlet-type conditions are modelled by a penalty term, as done for more complex models in [60],[61]. Let χ_c be the characteristic function of a given domain c, e.g., the circle in Figure 4.13. Let us consider the equation

$$\Delta u_\varepsilon = g + \frac{\chi_c}{\varepsilon}(u_\varepsilon - u_0). \tag{4.4}$$

We set $u_\varepsilon = u + \varepsilon\tilde{u}$ to derive the equations satisfied by u and $\tilde{u}$. By identifying the terms of the same order in ε, we have $\chi_c\,(u - u_0) = 0$ and $\Delta u = \chi_c\,\tilde{u}$. This formally implies that $u = u_0$ in the circle and $\Delta u = 0$ outside. Further analysis in [61] shows that $\|u_\varepsilon - u\|_2 = O(\sqrt{\varepsilon})$. Overall, u satisfies $u = u_0$ on the border of c with very good accuracy, taking $\varepsilon \approx 10^{-8}$.

The numerical discretization of the penalized model on an unfitted boundary will introduce an additional discretization error of order h, so that the scheme will be only first-order accurate. Second-order penalization can be obtained by extrapolation as shown in [62].

As explained, we impose Dirichlet conditions by a penalty term at the continuous level. We consider the exact solution $u_e(\boldsymbol{x}) = \sin((x_1 - 0.5)^2 + (x_2 - 0.5)^2)$, with the same square domain as before and a centred circle of radius 0.25. We take $u_0 = \sin\left((0.25)^2 + (0.25)^2\right)$ and $\varepsilon = 10^{-11}$. The grid is refined according to the distance function to the circumference (according to 4.1.3) in a layer of $\delta = 0.02$ on each side of the circle boundary; moreover the mesh is graded.

In Figure 4.13 we show the error distribution in the computational domain for tree level 7. Table 4.8 presents the error convergence. As expected, the penalized model is order one.

Tree level	L^∞	Order	L^2	Order
5	$1.392{\cdot}10^{-2}$		$3.216{\cdot}10^{-3}$	
6	$7.169{\cdot}10^{-3}$	0.971	$1.501{\cdot}10^{-3}$	1.071
7	$3.848{\cdot}10^{-3}$	0.932	$7.157{\cdot}10^{-4}$	1.048
8	$1.952{\cdot}10^{-3}$	0.986	$3.687{\cdot}10^{-4}$	0.970
9	$9.726{\cdot}10^{-4}$	1.003	$1.563{\cdot}10^{-4}$	1.18
10	$4.883{\cdot}10^{-4}$	0.996	$7.963{\cdot}10^{-5}$	0.981

Table 4.8: Error norms and order of the scheme.

To confirm the stiffness of the AMR approach on the penalized zone, we reproduce the test above with the exact solution on cells' centres when the cell splits the circumference. The order of this test is given in Figure 4.14, confirming the theoretical second-order expectations.

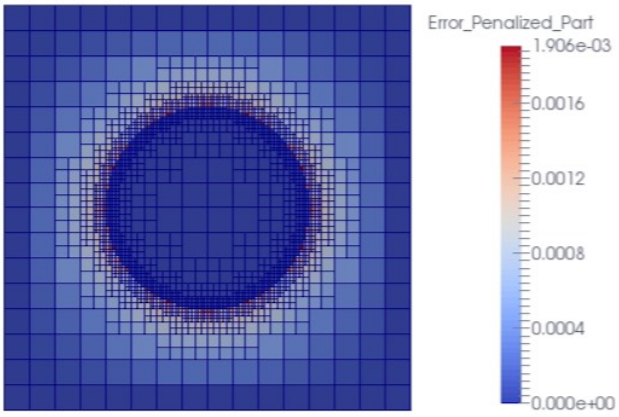


Figure 4.13: Error example, level of tree equal to 7 ($\Delta x = \Delta y = \frac{1}{2^7}$).

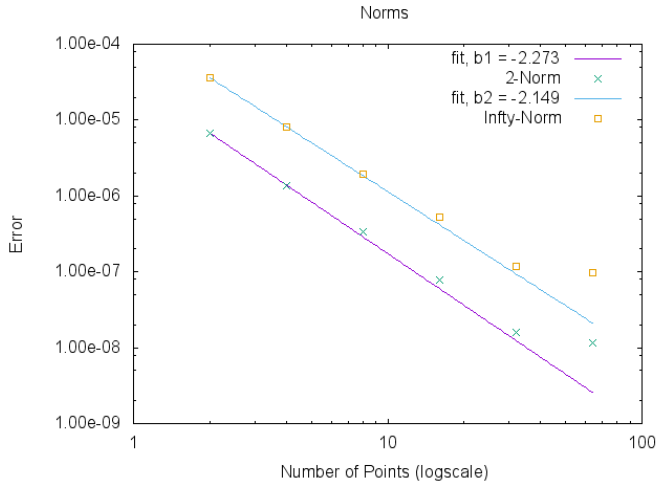


Figure 4.14: Exact boundary conditions convergence study.

4.2.1 Three Dimensional Results

The equivalent penalization tests have been reproduced in three dimensions. In this case, the radius of the sphere is 0.05 and the analytical solution is $u_e(\boldsymbol{x}) = \sin((x_1-0.5)^2+(x_2-0.5)^2+(x_3-0.5)^2)$. We focus the refinement on the penalized sphere (Fig. 4.16-4.17), as mentioned in 4.1.3, and we present the error study in Tables 4.9 and 4.10 for exact and approximated boundary, conditions respectively.

In figures 4.15-4.17 a section of the error is presented, while a zoom in near the penalized area is presented in Figure 4.16. For both figures, a part is more visible, retained to be able to distinguish the sections, and the

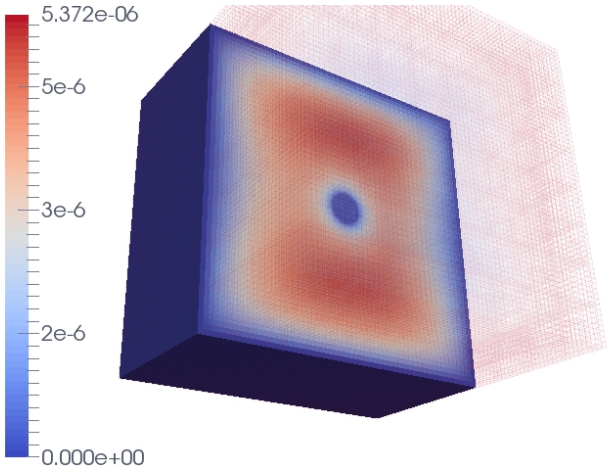


Figure 4.15: Exact boundary conditions convergence study. Error example of a penalization test (plane section). Radius of sphere: 0.05.

Level	Norm Infty	Norm 2	Mesh Points	Order L_2
6	$2.382{\cdot}10^{-4}$	$1.07{\cdot}10^{-4}$	2976	
7	$7.455{\cdot}10^{-5}$	$3.942{\cdot}10^{-5}$	10536	2.369
8	$2.432{\cdot}10^{-5}$	$1.196{\cdot}10^{-5}$	55672	2.15
9	$5.372{\cdot}10^{-6}$	$3.082{\cdot}10^{-6}$	380024	2.117
10	$1.387{\cdot}10^{-6}$	$8.050{\cdot}10^{-7}$	2826104	2.007
11	$3.524{\cdot}10^{-7}$	$2.057{\cdot}10^{-7}$	21907208	1.999

Table 4.9: Exact boundary conditions error convergence study.

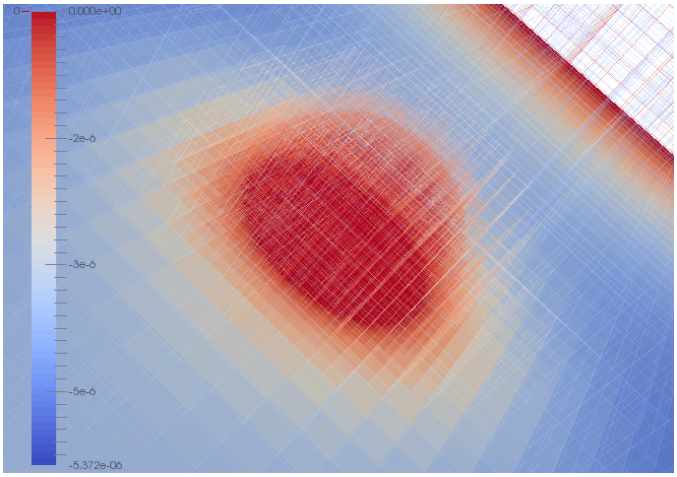


Figure 4.16: Zoom of error near the penalized spherical zone.

other one is an opaque wireframe of the mesh that allows us, precisely in

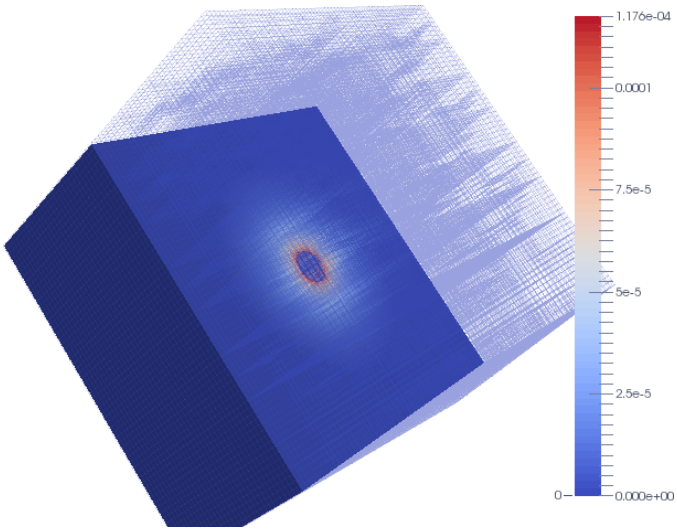


Figure 4.17: Approximated boundary conditions convergence study. Error example of a penalization test (plane section). Radius of sphere: 0.05.

Level	Norm Infty	Norm 2	Mesh Points	Order L_2
8	$1.31{\cdot}10^{-3}$	$1.509{\cdot}10^{-5}$	38256	-
9	$6.176{\cdot}10^{-4}$	$6.40{\cdot}10^{-6}$	330968	0.996
10	$3.508{\cdot}10^{-4}$	$3.719{\cdot}10^{-6}$	2646512	0.783
11	$1.802{\cdot}10^{-4}$	$1.979{\cdot}10^{-6}$	21205696	0.91

Table 4.10: Approximated boundary conditions error convergence study.

the second image, to distinguish the change of error in the proximity of the sphere. In fact, there is an obvious colour change where the mesh starts further refinement along the AMR, able to seize this discontinuity in the most efficient way possible, following the methodology and the principles already indicated in its two-dimensional section.

4.2.2 Uniform refinement and AMR for a multiscale problem

We investigate an idealized multiscale problem, taking the penalized subdomain much smaller than the rest; the configuration is given in Figure 4.18 and a zoom next to the circle Fig. 4.19. We consider the same exact solution above, the same square domain, but now the centred circle has radius 0.01. We take $u_0 = \sin\left((0.49)^2 + (0.49)^2\right)$ and $\varepsilon = 10^{-11}$. The grid is always refined according to the distance function to the border of the circle, we apply a distance to the border of δ equal two times the smallest octant' size for each level, following dynamically the equation $\delta = 2 * h_L$, $\forall M - 3 \leq L \leq M - 1$: if the octant, of size h_L, belongs to the circular ring considered (see 4.1.3) the cell splits into a new generation.

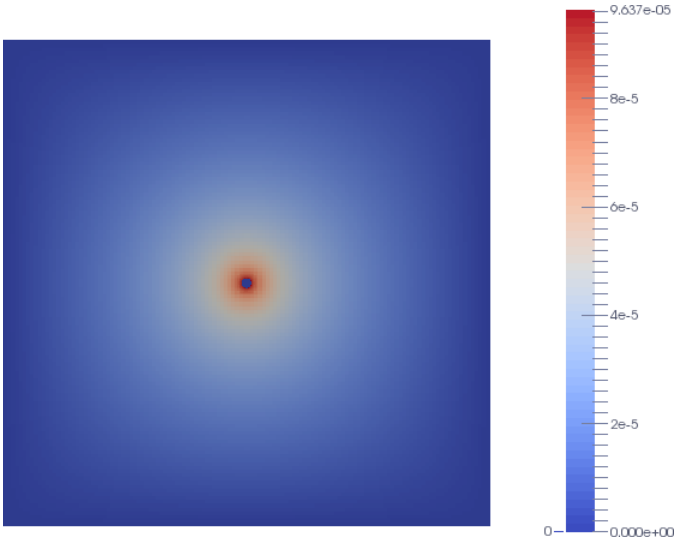


Figure 4.18: Error distribution when Dirichlet condition is imposed on a circle of radius 0.01.

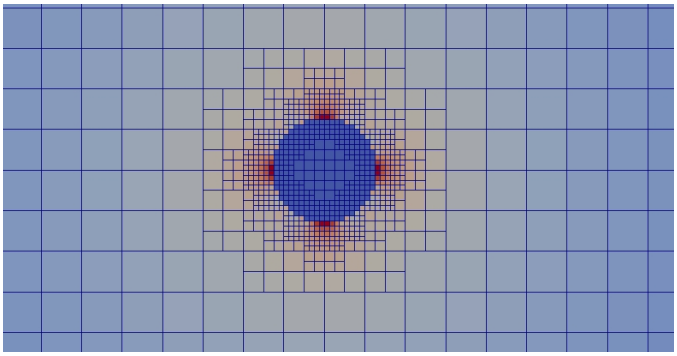


Figure 4.19: Zoom of the error next to the circle.

The aim of this section is to highlight the advantages of our AMR approach in comparison with uniform mesh resolutions. We remark that when a uniform stencil is performed we apply a standard five-point discretization for the Laplacian operator. The refinement of the uniform grid is chosen to give the same error compared to the non-uniform quadtree mesh at each level of the tree. For the uniform grid, we use two orderings of the unknowns. The first one is the standard ordering for a structured Cartesian grid, giving a pentadiagonal discretization matrix. We denote the uniform five-point structured scheme as *US* in the following Tables. For the sake of comparison, the same discretization stencil is used, but this time the unknowns are ordered with the Z-order (*UZ*). The discretization matrix contains the same weights of that of the US case, but it is not any-more

pentadiagonal. The non-uniform grid is obtained by mesh refinement next to the circle with three levels of jump between the maximal and the minimal depth of tree, enforcing balancing constraints through faces. The results of the finite-difference scheme on the non-uniform mesh are denoted by *AMR*. The Krylov space used for this test is BCGS with ASM preconditioning and ILU sub-preconditioner.

In tables 4.11, 4.12 and 4.13, the first line reports the results for level 7 of grid refinement ($h = \frac{1}{2^7}$) and the last one for level 15. For the *AMR* case, the first three lines do not exist because of the non-uniform refinement (the largest grid is at tree level 10, where the minimal depth of the tree is 7).

Tree's Level	L^2 US	L^∞ US	L^2 UZ	L^∞ UZ	L^2 AMR	L^∞ AMR
7	$5.25{\cdot}10^{-6}$	$3.49{\cdot}10^{-5}$	$9.30{\cdot}10^{-6}$	$5.72{\cdot}10^{-5}$	–	–
8	$4.15{\cdot}10^{-6}$	$2.79{\cdot}10^{-5}$	$1.31{\cdot}10^{-6}$	$1.29{\cdot}10^{-5}$	–	–
9	$8.96{\cdot}10^{-7}$	$9.20{\cdot}10^{-6}$	$2.62{\cdot}10^{-6}$	$1.81{\cdot}10^{-5}$	–	–
10	$9.56{\cdot}10^{-7}$	$9.09{\cdot}10^{-6}$	$1.08{\cdot}10^{-6}$	$1.08{\cdot}10^{-5}$	$3.63{\cdot}10^{-6}$	$1.07{\cdot}10^{-5}$
11	$6.57{\cdot}10^{-7}$	$5.75{\cdot}10^{-6}$	$7.71{\cdot}10^{-7}$	$8.01{\cdot}10^{-6}$	$7.47{\cdot}10^{-7}$	$7.99{\cdot}10^{-6}$
12	$3.7{\cdot}10^{-7}$	$4.16{\cdot}10^{-6}$	$3.05{\cdot}10^{-7}$	$2.87{\cdot}10^{-6}$	$2.08{\cdot}10^{-7}$	$2.87{\cdot}10^{-6}$
13	$1.64{\cdot}10^{-7}$	$2.05{\cdot}10^{-6}$	–	–	$1.08{\cdot}10^{-7}$	$1.49{\cdot}10^{-6}$
14	$6.36{\cdot}10^{-8}$	$9.34{\cdot}10^{-7}$	–	–	$5.58{\cdot}10^{-8}$	$7.33{\cdot}10^{-7}$
15	$1.13{\cdot}10^{-8}$	$3.26{\cdot}10^{-7}$	–	–	$1.16{\cdot}10^{-8}$	$3.81{\cdot}10^{-7}$

Table 4.11: Errors for the configuration shown in Fig. 4.18. US stands for the uniform structured grid, UZ for uniform mesh with Z-order and AMR for the adaptive mesh with three jumps of level.

In Table 4.11, we present the error norms for the configuration shown in Figure 4.18. This Table shows that the uniform grid refinement is such that the errors are approximatively the same as those for the non-uniform grid, where they are at same maximal depth of refinement. The slight differences between US and UZ are due to the iterative linear solver, because of the different numbering, even though the grid is exactly the same.

In Table 4.12, we show the number of grid points for the corresponding error levels in Table 4.11 (the grids US and UZ have of course the same number of cells). As expected, the number of grid points are strongly reduced in the adaptive mesh method. At level 15, the AMR grid has approximatively 220 times fewer grid points.

In Table 4.13, we report the total CPU time required to solve the linear problem. All the tests presented in this section have been solved using 96 cores on 4 nodes. The time in seconds refers to the elapsed time of the KSPSolve() module of PETSc. The computational time needed for the AMR is one to two orders of magnitude less compared to the US grid. Because of the numbering, the UZ case is way more expensive.

Tree's Level	US/UZ	AMR
7	16384	–
8	65536	–
9	262144	–
10	1048576	4900
11	4194304	19012
12	16777216	76036
13	67108864	302776
14	268435456	1214512
15	1073741824	4863616

Table 4.12: Number of grid points.

Tree's Level	US	UZ	AMR
7	$3.533{\cdot}10^{-2}$	$8.299{\cdot}10^{-2}$	–
8	$2.641{\cdot}10^{-2}$	$2.926{\cdot}10^{-1}$	–
9	$6.564{\cdot}10^{-2}$	$1.812{\cdot}10^{0}$	–
10	$5.101{\cdot}10^{-1}$	$1.715{\cdot}10^{1}$	$3.796{\cdot}10^{-2}$
11	$3.764{\cdot}10^{0}$	$2.078{\cdot}10^{2}$	$1.109{\cdot}10^{-1}$
12	$2.837{\cdot}10^{1}$	$2.387{\cdot}10^{2}$	$4.104{\cdot}10^{-1}$
13	$1.904{\cdot}10^{2}$	–	$2.428{\cdot}10^{0}$
14	$1.241{\cdot}10^{3}$	–	$2.23{\cdot}10^{1}$
15	$7.647{\cdot}10^{3}$	–	$2.710{\cdot}10^{2}$

Table 4.13: Times (in seconds) to solve the linear problem.

Of course, the results of this section depend on the configuration studied, e.g., the ratio between the square side and the circle diameter. The point we make is that, even though Z-order may significantly reduce performance of linear solvers, the number of grid points is reduced to an extent that makes the solution by far faster, as well to justifies its use in these kinds of applications.

4.3 Diffusion coefficient discontinuity

We consider in this section the equations:

$$\kappa(\boldsymbol{x})\Delta u(\boldsymbol{x}) = -1.0, \quad \text{in } \Omega = [0,1]\times[0,1] = G \cup S, \tag{4.5a}$$

$$[\kappa(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})] = 0, \text{ on } \gamma. \tag{4.5b}$$

The region S is defined by a circle of radius $R = 0.25$.

In many applications, the diffusion coefficient $\kappa(x)$ can abruptly vary across an interface from, say, α to β, two positive constants. As it is well

known, in the limit case the solution and the normal fluxes are continuous at the interface. We model these problems by the regularised diffusion function

$$\kappa(\boldsymbol{x}) = \alpha + (\beta - \alpha)\left(\frac{\tanh\left(\sigma \cdot \Phi(\boldsymbol{x})\right) + 1}{2}\right), \tag{4.6}$$

where $\Phi(x)$ is the distance function with respect to the interface of discontinuity and σ is the regularisation parameter. The distance function is obtained by solving $|\nabla\Phi(x)| = 1$ together with a homogeneous Dirichlet boundary condition on the interface.

Overall, also in this case we expect first-order convergence for the discontinuous coefficient case. We will recover accurate enough results using local grid refinement thanks to the distance function to the boundary.

We consider now the full problem of equations 4.5. The diffusion coefficient $\kappa(x)$ is piecewise constant. It is equal to one in G and one hundred in S. The parameters of equation (4.6) relative to this test are:

$$\alpha, \beta \text{ such that } \alpha + \frac{1}{2}(\beta - \alpha) = 49.5 + 1, \text{ and } \frac{\beta - \alpha}{2} = 49.5,$$

$$\sigma = 100.$$

The limit problem solution $u_e(x, y)$ (Fig. 4.20) with appropriate boundary conditions on the sides of the square is:

$$u_G(x, y) = \frac{1}{8} - \frac{1}{4}((x - 0.5)^2 + (y - 0.5)^2) \quad (x, y) \in G, \tag{4.7a}$$

$$u_S(x, y) = (\frac{1}{8} - \frac{R^2}{4}(1 - \frac{1}{\kappa_S})) - \frac{1}{4\kappa_S}((x - 0.5)^2 + (y - 0.5)^2) \quad (x, y) \in S. \tag{4.7b}$$

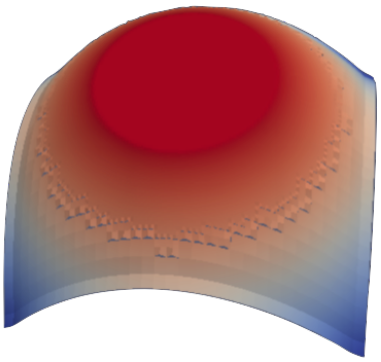


Figure 4.20: Analyitical test function.

We depict the convergence results obtained referring to this specific solution in Tables 4.14-4.15.

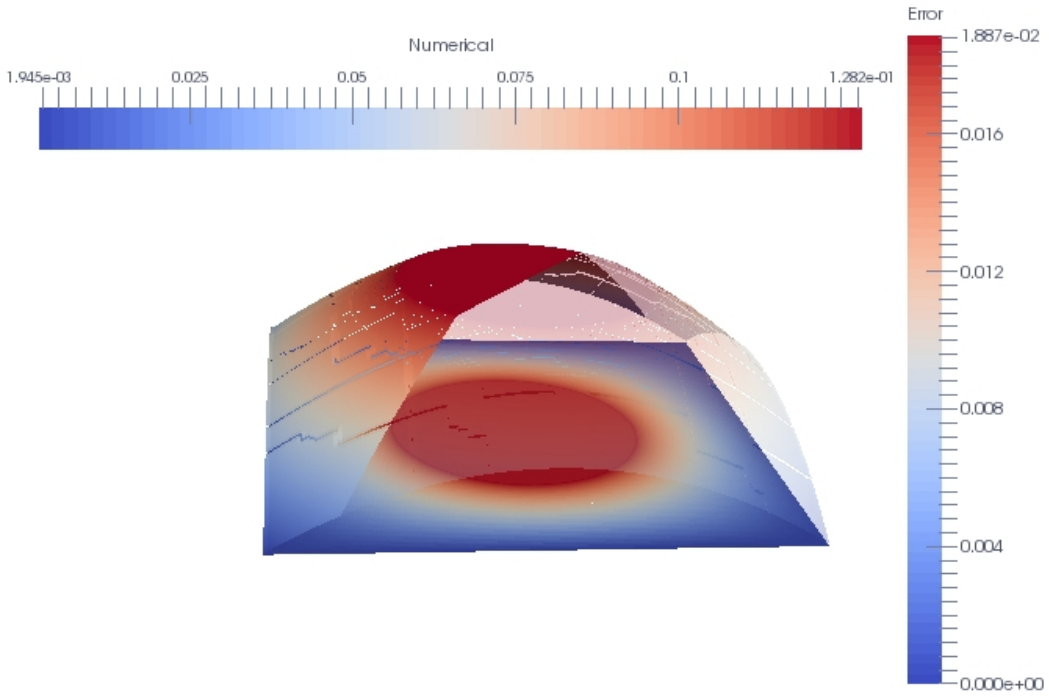


Figure 4.21: Numerical result obtained by AMR: numerical solution plane section, with a projection of the error.

Tree's Level	Norm Infty	Norm 2	Mesh Points
7	$2.406 \cdot 10^{-2}$	$1.694 \cdot 10^{-1}$	3784
8	$2.438 \cdot 10^{-2}$	$1.74 \cdot 10^{-1}$	14968
9	$3.861 \cdot 10^{-2}$	$2.778 \cdot 10^{-1}$	51472
10	$1.866 \cdot 10^{-2}$	$1.343 \cdot 10^{-1}$	112684
11	$2.878 \cdot 10^{-3}$	$1.985 \cdot 10^{-2}$	228484
12	$6.602 \cdot 10^{-4}$	$3.821 \cdot 10^{-3}$	465028
13	$1.543 \cdot 10^{-4}$	$1.03 \cdot 10^{-3}$	1124968

Table 4.14: Convergence results. Difference between AMR along the mollification and external mesh: five levels.

The test in Table 4.14 is built on a mesh that focuses on mollification according to the following criteria (Fig. 4.23):

- the maximal depth of the tree is fixed at value M from 7 to 13 for presented results;
- the squared domain is uniformly refined until level $M - 3$;
- from $M-3$ to M, the mollified function (4.6) is valued on each octant.

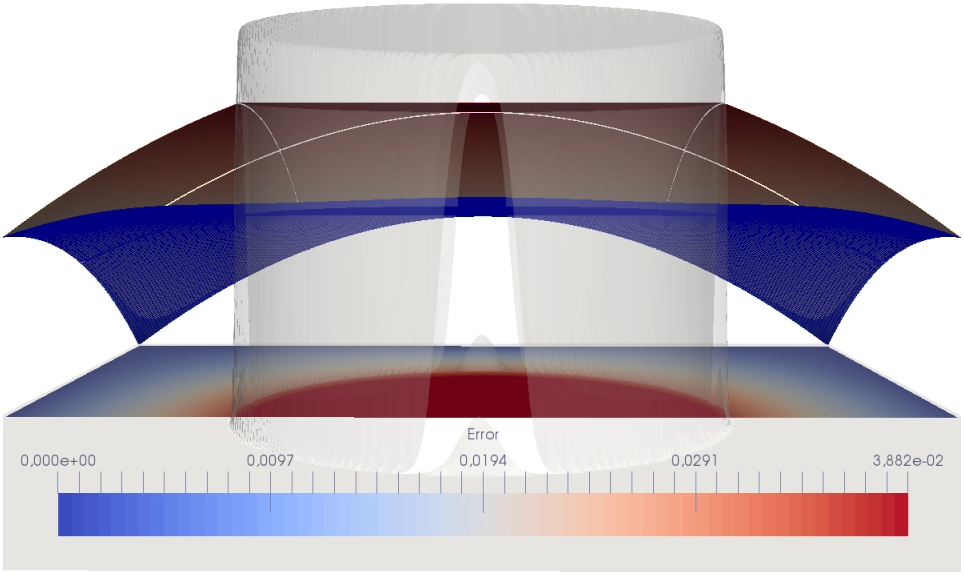


Figure 4.22: Numerical result obtained on uniform mesh: numerical solution plane section (blue) and analytical one (upper), with a projection of the error (bottom) comparable in front of the conductivity term κ (earl grey).

If the octant lies on the regularization zone, that means $\nabla\kappa(\boldsymbol{x}) \neq 0$, and it splits in four children;

- balance constraints are applied on the jump zones.

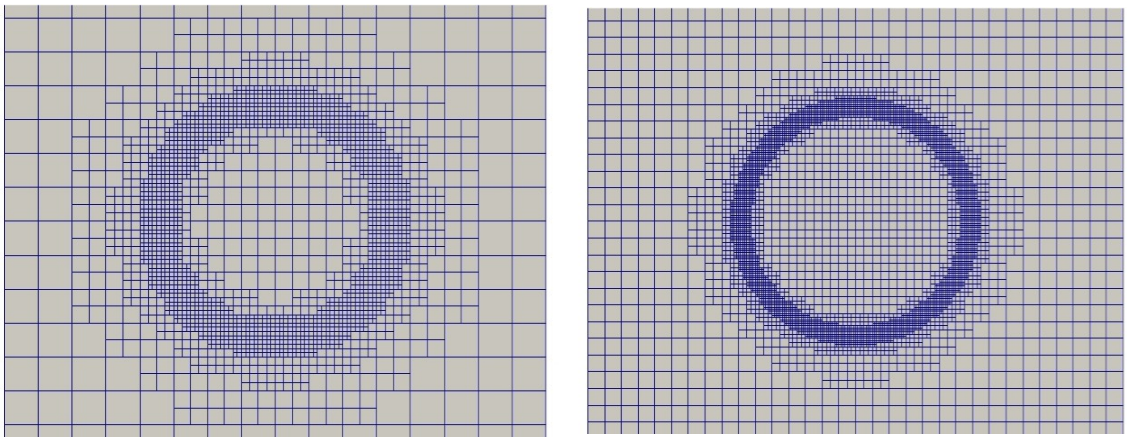


Figure 4.23: Levels 7 and 8 of the AMR along the mollification with three levels of jump admitted.

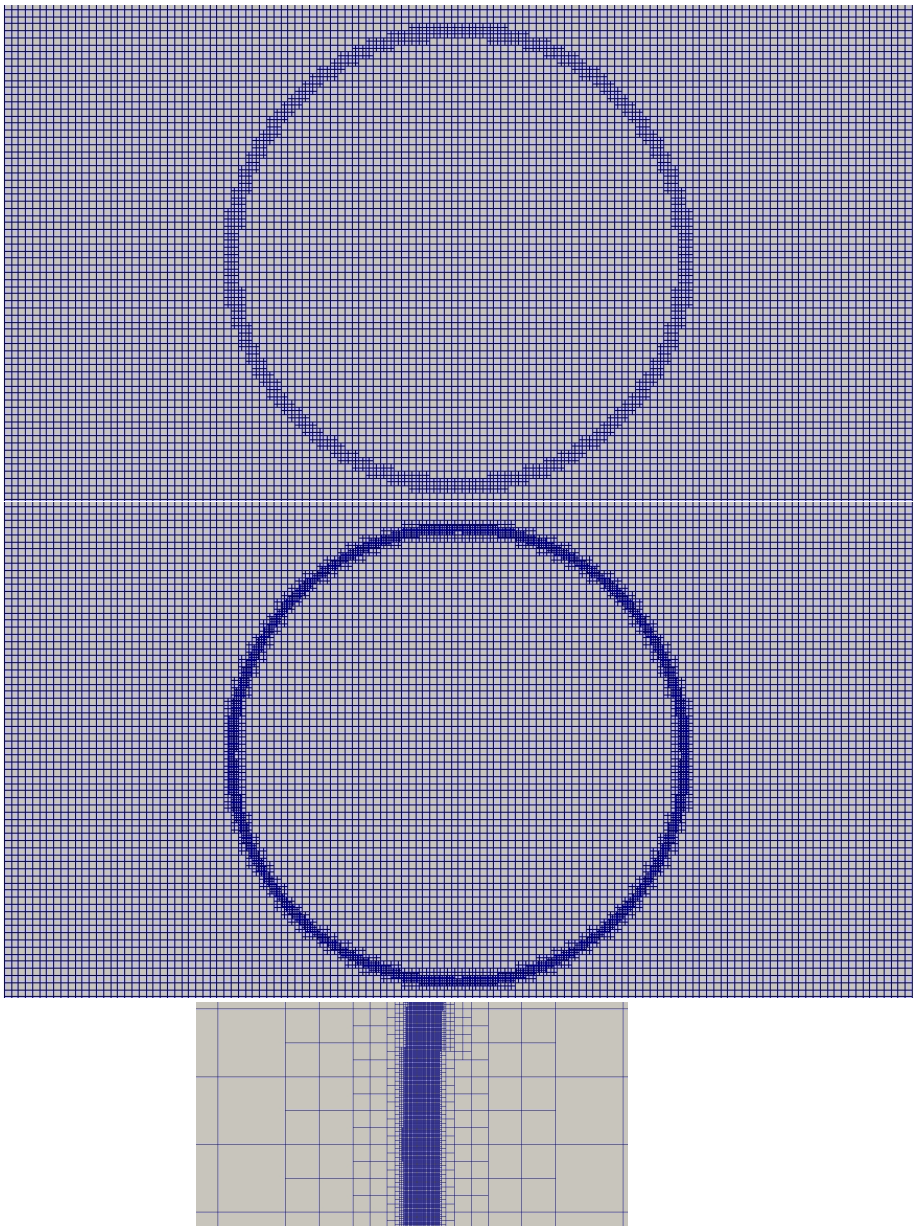


Figure 4.24: Levels 8 and 9 of the AMR along the mollification with uniform mesh outside at level 7 (top). Zoom of level 13 on bottom.

This refinement is compared to another one (Fig 4.24) connected to Table 4.15 where the refinement follows:

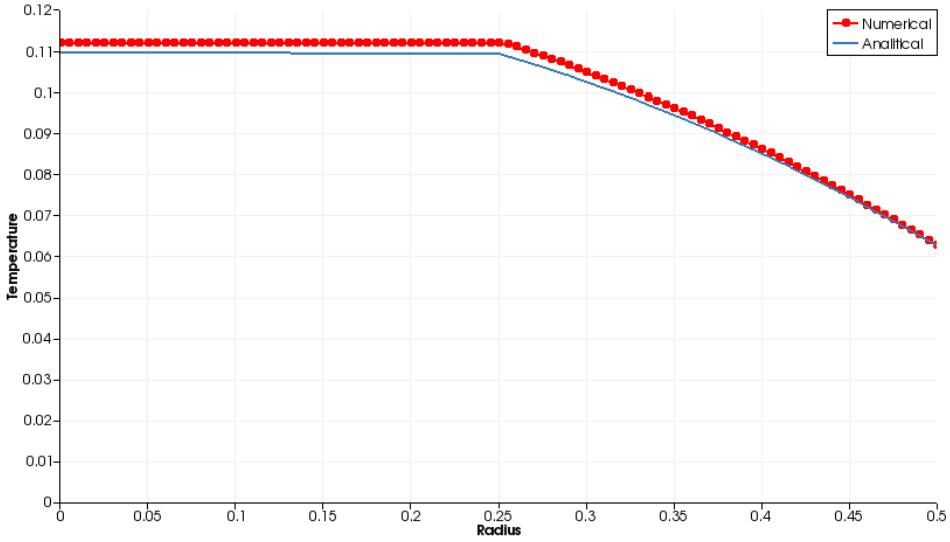
- the maximal depth of the tree is fixed at value M from 7 to 13;
- the squared domain is uniformly refined until level 7 (16384 octants);
- from 7 to M, the mollified function (4.6) is valued on each octant. If the octant lies on the regularization zone, that means $\nabla\kappa(\boldsymbol{x}) \neq 0$, and it splits in four children.
- balance constraints are applied on the jump zones.

Tree's Level	Norm Infty	Norm 2	Mesh Points
7	$2.63 \cdot 10^{-2}$	$1.895 \cdot 10^{-1}$	16384
8	$2.567 \cdot 10^{-2}$	$1.848 \cdot 10^{-1}$	30868
9	$3.901 \cdot 10^{-2}$	$2.815 \cdot 10^{-1}$	59896
10	$1.887 \cdot 10^{-2}$	$1.362 \cdot 10^{-1}$	117796
11	$2.909 \cdot 10^{-3}$	$2.014 \cdot 10^{-2}$	233512
12	$6.602 \cdot 10^{-4}$	$3.821 \cdot 10^{-3}$	465028
13	$1.535 \cdot 10^{-4}$	$1.028 \cdot 10^{-3}$	1090300

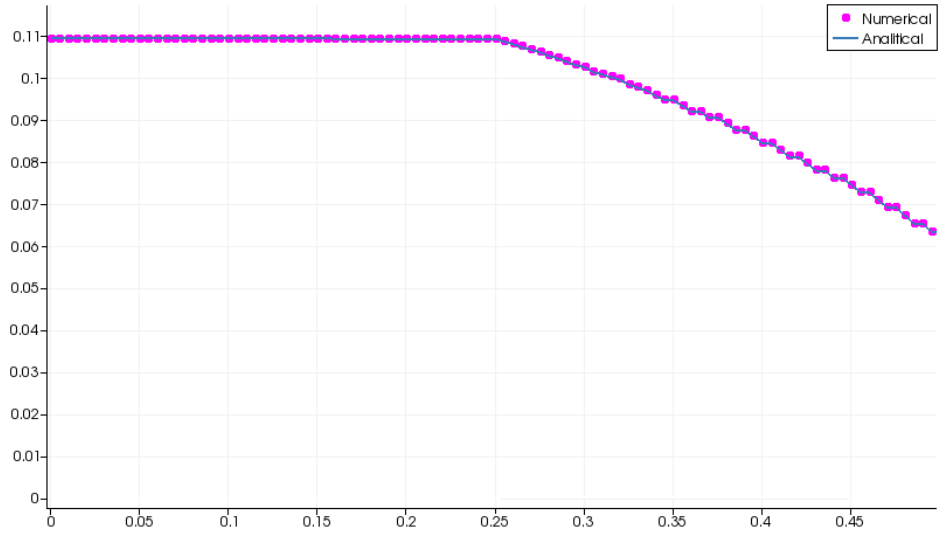
Table 4.15: Convergence results. AMR along the mollification outside level 7 is fixed.

The convergence result presented in Table 4.15 is built on a grid that increase the refinement level by level on the mollification part and stands constant outside at level seven of the tree. This choice is compared to Table 4.14, where the mesh is splitting in both directions, retaining a constant difference of level between the maximal and the minimal depth of the tree. We evidence that most of the error is focused on the mollification part while the external part refinement does not have a strong influence on the global results. This point allowed us to reach a certain gain of memory on degrees of freedom with a local focus on the part concerning the discontinuities. The first line of each Table is the level 7 case; it means that for the second case the first line is an uniform mesh and its result is comparable to the AMR one in Table 4.14.

In Figure 4.25, once the memory gain has been presented, we focus on the accuracy. The uniform grid covers the domain with 1048576 degrees of freedom, the level of the tree is ten, and the results employ $2,6 \cdot 10^2$MB. The AMR concerned in this test is the same presented in Table 4.15, where the level of the tree is fixed at seven outside, with finer levels until thirteen on the mollification; 1090300 degrees of freedom are involved and the results employ $2,3 \cdot 10^2$MB. Being equal the occupied memory and the order of degrees of freedom, this comparison highlights that the AMR approach sticks better



(a) Uniform mesh. 1048576 degrees of freedom.



(b) AMR approach. 1090300 degrees of freedom.

Figure 4.25: Comparison between uniform and AMR resolutions

on the solution; on the other hand, where the refinement is less fine the function appears piecewise continuous, but it follows the discontinuity with a finer approach. Then a sensitive precision is guaranteed on the entire computational domain.

Analytical Proof

We briefly demonstrate that the analytical solution (4.7) satisfies the problem (4.5a)-(4.5b) for piecewise constant diffusion coefficient. Let κ_i be (κ_G, κ_S), remembering that $\kappa_G = 1$.

$$\partial_x u_e(x,y) = -\frac{1}{2\kappa_i}(x-0.5), \quad \partial_y u_e(x,y) = -\frac{1}{2\kappa_i}(y-0.5)$$

$$\partial_{xx} u_e(x,y) = -\frac{1}{2\kappa_i}, \quad \partial_{yy} u_e(x,y) = -\frac{1}{2\kappa_i}$$

$$\Downarrow$$

$$\kappa_i \Delta u_e(x,y) = \kappa_i(\partial_{xx} u_e(x,y) + \partial_{yy} u_e(x,y)) =$$

$$= \kappa_i(-\frac{1}{2\kappa_i} - \frac{1}{2\kappa_i}) = -1 \Rightarrow (4.5a). \qquad \square$$

We point out that the two paraboloids u_G and u_S are symmetrical with respect to the curve γ, so we can calculate the normal derivative on a generic point $P(0.25, 0.5)$ of the circumference and consider the calculation that is valid for each point of the curve. On P the normal unit vector $\boldsymbol{n}$ has opposite directions with respect to γ. We evaluate:

$$[\kappa(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})]_\gamma|_P = \kappa_G \partial_{\boldsymbol{n}} u_G(\boldsymbol{x})|_P - \kappa_S \partial_{\boldsymbol{n}} u_S(\boldsymbol{x})|_P =$$

$$= \kappa_G((\partial_x u_G, \partial_y u_G) \cdot \boldsymbol{n}_G)|_P - \kappa_S((\partial_x u_S, \partial_y u_S) \cdot \boldsymbol{n}_S)|_P =$$

$$= \kappa_G((-\frac{1}{2\kappa_G}(x-0.5), -\frac{1}{2\kappa_G}(y-0.5)) \cdot (-1,0))|_P -$$

$$\kappa_S((-\frac{1}{2\kappa_S}(x-0.5), -\frac{1}{2\kappa_S}(y-0.5)) \cdot (1,0))|_P =$$

$$= \kappa_G(-\frac{1}{2\kappa_G}(x-0.5))|_P - \kappa_S(-\frac{1}{2\kappa_S}(x-0.5))|_P = 0 \Rightarrow (4.5b).$$

$\square$

4.4 The parallel code

In this section, some outlines of the code are presented. We explained before that this work was conceived to be optimal in parallel; however, we can split the code in two main parts: the first one where the cores are almost independent of each other and the second one with greater communication.

First of all, the guidelines for the mesh construction are dictated by the problem:

1. set the max depth of the tree;
2. creation of the AMR with balancing constraints;
3. load balance on the cores.

Step one is user defined; in fact, the only variable required to execute the code from command line is the depth of the tree. Then, the set of balancing following the dimension of the problem and the type of mesh required is automatic in PABLO. During this construction, a balancing process is made so that there is a fair distribution of the octants between the process and a spatial contiguity as possible, as discussed above in the chapter 2.

4.4.1 The weights calculation

After establishing the refinement and the balancing on nodes, each process traverses the octants that belong to him locally once, and for each of them:

- a first study of neighbourhood allows us to build the *identification key* where required;
- using the key (the neighbours for unbalanced meshes) the vectors of axial distances from the octant to its neighbours are built;
- the vectors of axial distances are applied to compute the optimal local minimization problem in order to the configuration (if the local configuration through faces is uniform the five points stencil is applied);
- the linear problem is solved with PETSc, obtaining the weights a_i, they are stored on a vector of doubles;
- a second analysis of neighbourhood obtains the global indices of the octants, and the integer vector of their values is devised;
- the vector of indices and the vector of weights are both used to add the global values in the *Mat* object PETSc which describes the complete physical model to solve.

The interprocess communication in this preliminary phase is provided during the neighbourhood analysis; indeed during the local investigation (through faces, edges and vertices) if a neighbour appertains to another process is said **ghost** for that octant. When a ghost neighbour is evaluated, PABLO allows us to obtain easily the informations contained; this communication is thus concealed to the user, who does not need to directly request information at the adjacent processes. On the other hand, in each process, at the beginning of the calculation, some data regarding the geographically adjacent ghost points are retained so that these communications, to lighten time calculations, are limited only to cases of real need, whereas access to that information through of the local vectors is faster.

4.4.2 Parallel solution of PDEs

We make use of PETSc [63] for the numerical solution of partial differential equations; this library is conceived for these kind of problems on high-performance computers. The PorTable, Extensible Toolkit for Scientific Computation (PETSc) is a suite of data structures and routines that provides the building blocks for the implementation of large-scale application codes on parallel (and serial) computers. PETSc uses the MPI standard for all message-passing communication.

In the weights calculation is contained a part that build the *Mat* object PETSc, called M, which resumes the numerical operator. In this second part of the code, the interprocess communication is handled by its routines through which we are able to:

- pre-processing phase: we define the right hand side terms on a *Vec* object $\boldsymbol{f}$;
- define the Krylov (KSP) and preconditioner methods adapted to the problem;
- solve the problem $M\boldsymbol{x} = \boldsymbol{f}$;
- post-processing phase: we extract $\boldsymbol{x}$, compute errors and results.

The methods may vary depending on the problem; generally the chosen Krylov subspace is Flexible GMRES (FGMRES) while we try to optimize the preconditioning. For example, in the test case in 4.4.3 the resolution choices are:

```
-pc_type asm
-pc_asm_overlap 8
-sub_pc_type ilu
-sub_pc_factor_levels 8
```

which define the overlapping between the processes of an Additive Schwartz preconditioner with ILU sub-preconditioner.

KSP Outlines

A lot of computational problems require solving linear systems:

$$A\boldsymbol{x} = \boldsymbol{b}. \tag{4.9}$$

When these systems have small sizes in general they can be approached using direct solvers; however, to solve partial differential equations generates A arbitrarily large. For these reasons, iterative resolution methods have become necessary in modern computations. The combination of a Krylov subspace method and a preconditioner has recently spread in numerical codes for

the iterative solution of linear systems [64]. The Krylov subspace methods produce approximations $\boldsymbol{x}_n$ to $A^{-1}\boldsymbol{b}$ such that:

$$\boldsymbol{x}_n \in \boldsymbol{x}_0 + \mathcal{K}_n(\boldsymbol{r}_0, A),$$

where $\boldsymbol{x}_0 \in \mathbb{C}^N$, with N number of unknowns, is any initial guess for (4.9), $\boldsymbol{r}_0 = \boldsymbol{b} - A\boldsymbol{x}_0$, the residual vector, and $\mathcal{K}_n(\boldsymbol{r}_0, A) = \{\phi(A)\boldsymbol{r}_0 | \phi \in \mathcal{P}_{n-1}\}$ the Krylov subspace at n-step generated by $\boldsymbol{r}_0$.

For an efficient solution, it is necessary to use preconditioning methods. Let M be a non-singular matrix in $\mathbb{R}^{N\times N}$, which approximates A and decomposes in $M = M_1 M_2$. The preconditioned method solves:

$$A'\boldsymbol{x}' = \boldsymbol{b}', \tag{4.10}$$

where $A' = M_1^{-1} A M_2^{-1}$, $\boldsymbol{b}' = M_1^{-1}\boldsymbol{b}$, $\boldsymbol{x}' = M_2\boldsymbol{x}$. The problems residuals are related by

$$\boldsymbol{x}_n = M_2^{-1}\boldsymbol{x}'_n, \quad \boldsymbol{r}_n = M_1\boldsymbol{r}'_n.$$

Since the rate of convergence of Krylov projection methods for a particular linear system is strongly dependent on its spectrum, preconditioning is typically used to alter the spectrum and hence accelerate the convergence rate of iterative techniques. This concludes the advantages to properly adapt the Krylov method to sub-preconditioner for our purpose.

Remark 4.4.1 (About FGMRES) *The choice of the iterative method is not an immediate process, not even the suitable type of preconditioning. In this work, the use of FGMRES is preferred for its stability (an accurate analysis of FGMRES performances on ill-posed problems is given [65]). However, for indefinite and highly non-symmetric matrices (which is our case) the performance of a given preconditioner can be unpredictable as proved by Saad [66], who presented this algorithm in 1993.*

The main difference between the standard GMRES and the flexible one is that in each iteration the preconditioned vectors obtained solving (4.10) *are stored in memory and the solution is updated using them. Moreover, it allows larger flexibility in the choice of solution subspace than GMRES , but the memory increase for these spaces is offset by more speed convergence for a solution.*

Another advantage of the flexible GMRES in this work is its adaptivity to mixed preconditioners to solve a given problem.

4.4.3 Code Scalability

To test the code scalability, we reproduced the test in section 4.3, with equations:

$$\kappa(\boldsymbol{x})\Delta u(\boldsymbol{x}) = -1.0, \quad \text{in } G \cup S, \tag{4.11a}$$

$$[\kappa(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})] = 0, \text{ on } \gamma. \tag{4.11b}$$

$\kappa(\boldsymbol{x})$ is piecewise constant from 1 to 100, following the formulation (4.6). The analytical solution is composed of the two paraboloids in (4.7) intersecting on the circumference with radius 0.25.

The test investigated is on 228484 degrees of freedom with an adaptive mesh refinement across the mollified discontinuity, where the depth of the tree is 11 and the smallest cells have size $\Delta x = \Delta y = 0.000488281 = 1/2^{11}$.

The test has been executed on PlaFRIM; each node used has the following properties:

- 2 Dodeca-core Haswell Intel ®Xeon ®E5-2680 v3 @ 2,5 GHz
- 128Go de RAM
- Infiniband QDR TrueScale: 40Gb/s
- Ethernet : 10Gb/s

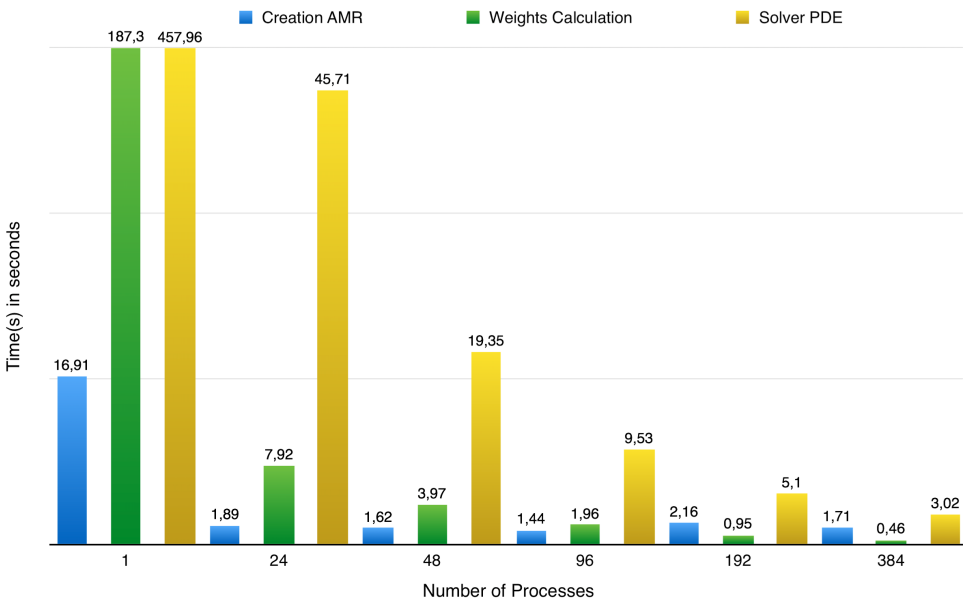


Figure 4.26: Time comparison for different parallel cases (strong scalability)

Figure 4.26 shows the calculation time for the three main operations of the execution. Except for the first diagram showing the serial results, the nodes are optimized using all the available processes (24 for each node). This means, for example, that 384 processes are distributed on 16 nodes, fully occupied in computation.

The time comparison highlights that the creation of the mesh does not have significant parallelism advantages. This result is due to the fact that the load balance on nodes is not recursive on each level of the tree. Meanwhile

without intermediate balance, we can see an evident worsening on the serial case (first column). On the other hand, we can see, for the two main code operations, that as the number of processes is doubled, the execution time is approximately halved.

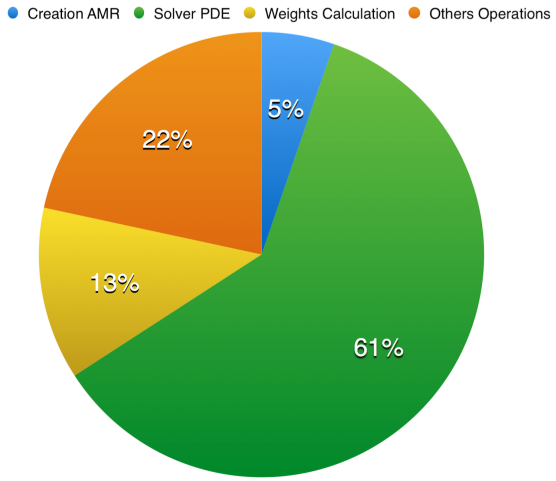


Figure 4.27: Time partition for a complete execution. 48 cores distributed on 2 nodes. Total execution time 29 sec(s)

In Figure 4.27, a time partition is presented. This execution concerns 48 processes (2 nodes). We can highlight that the most expensive operation is the parallel solver, while the weights calculation occupies only 13% of the complete execution. With *Others Operations* are implied the data constructions necessary for the complete resolution (right hand side imposition, boundary conditions, and go on) that employ a negligible amount of time.

In Figure 4.28, a weak scalability result is presented. To achieve this test, solving still equations (4.11a)-(4.11b), we applied a uniform mesh over the domain and we considered 4096 degrees of freedom for each processor.

We quadruple the degrees of freedom and the processors at the same time. We expect to preserve the resolution times for the two main blocks, and we can actually see how the process on the octants (in light blue) is performing, which is almost constant. This test allows us to check the good scalability on the octants in weighing and computing the matrix operations. On the other hand the Laplacian resolution time grows by a factor of 0.5, but we remember that the resolution time is dependent of the degrees of freedom.

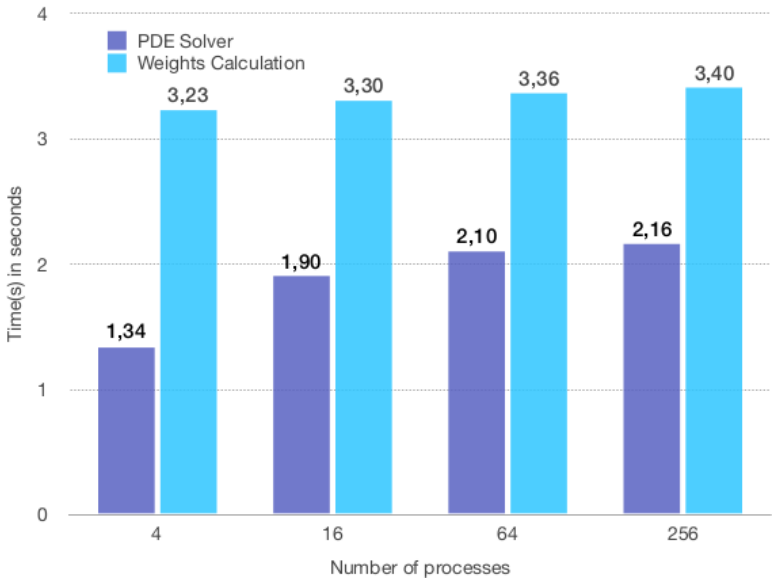


Figure 4.28: Time comparison for different parallel cases (weak scalability)

Chapter 5

The Heat Equation Applications

We introduced the phase-changing material (PCM) on which we would like to apply our numerical method. We cited as example that the physical interpretation of the Laplace equation with Dirichlet boundary conditions is a heat application through the surface of a domain which will reach a temperature up to a steady state inside; meanwhile for Neumann boundary conditions, we face an isolated boundary (see [1]-[2]). In this chapter, we explain the problem and we study in detail the real applications we have referred to. It is initially proposed to derive the heat and diffusion equation and then more meticulously analyse the properties of the PCM by presenting some numerical tests. The concluding part of this thesis therefore explains the bases of application of our method in real and environment applications.

5.1 Heat Conduction

Heat is a form of energy that moves from warmer regions to cooler areas of a body. The study of heat conduction is mostly based on the analysis of the temperature distribution, which we will denote with $u(\boldsymbol{x})$, where $\boldsymbol{x}$ represents the spatial and time coordinates. It is possible to determine the heat flow in a region from $u(\boldsymbol{x})$ often subject to appropriate boundary and initial conditions.

We suppose a *homogeneous* and *isotropic* conducting material; thus, the variables governing the heat conduction phase are independent from their spatial position. Let ΔQ be the amount of heat $u_{n+1} - u_n$ in time Δt, where $u_n = u(x, y, z, t_n)$ represents the generic temperature on a point at time t_h. The amount of heat in a volumetric domain, Ω with area $A = A(\Omega)$, is proportional to the rate of decrease in temperature u. In other words, given $\boldsymbol{q}$, a vector of heat flow, we can resume this dependency with:

$$\boldsymbol{q} = -\kappa \nabla u, \tag{5.1}$$

where κ is the *thermal conductivity* of the material. The total heat flux in unit time, leaving the surface across A is so obtained:

$$\int\int_A \boldsymbol{q} \cdot \, d\boldsymbol{A} = \int\int_A (-\kappa \nabla u) \cdot \boldsymbol{n} \, dA. \tag{5.2}$$

The quantity of heat entering the surface is given by:

$$\int\int_A (\kappa \nabla u) \cdot \boldsymbol{n} \, dA = \int\int\int_\Omega \nabla \cdot (\kappa \nabla u) \, d\Omega. \tag{5.3}$$

The heat dQ contained on a part of volume $d\Omega$, with ρ the density of the material and c the specific heat is:

$$dQ = (\rho c) u \, d\Omega. \tag{5.4}$$

Meanwhile, the increase of heat in the conducting body is:

$$\frac{dQ}{dt} = \frac{d}{dt} \int\int\int_\Omega (\rho c) u \, d\Omega = \int\int\int_\Omega (\rho c) \frac{\partial u}{\partial t} \, d\Omega. \tag{5.5}$$

Remembering that we supposed an homogeneous material, κ is constant in space; thus we obtain:

$$\int\int\int_\Omega \kappa \nabla \cdot (\nabla u) \, d\Omega = \int\int\int_\Omega (\rho c) \frac{\partial u}{\partial t} \, d\Omega \tag{5.6}$$

$$\Downarrow$$

$$\int\int\int_\Omega (\kappa \nabla \cdot (\nabla u) - (\rho c) \frac{\partial u}{\partial t}) \, d\Omega = 0 \tag{5.7}$$

$$\Downarrow$$

(since we used an arbitrary volume $d\Omega$)

$$\kappa \nabla \cdot (\nabla u) = \kappa \Delta u = \rho c \frac{\partial u}{\partial t}, \tag{5.8}$$

thus, we obtain the heat conduction equation, developed by Joseph Fourier, *i.e.*

$$\alpha \Delta u = \frac{\partial u}{\partial t}, \tag{5.9}$$

where $\alpha = \frac{\kappa}{\rho c}$ is named *thermal diffusivity* of the conducting material. The thermal diffusivity has an intrinsic relationship with heat propagation: the higher its value, the faster the propagation will be.

Remark 5.1.1 *When the material reaches the steady state temperature, the problem reduces to Laplace's equation:*

$$\Delta u = 0.$$

Boundary Conditions: Physical Properties

Given the equation (5.9) over a domain Ω that represents the conducting body, different cases are possible:

Dirichlet boundary condition. $u = u_D$, on $\partial\Omega$, indicates a boundary surface is maintained at constant temperature, as example a fixed source that influences the surface.

Neumann boundary condition. $\frac{\partial u}{\partial \boldsymbol{n}} = u_N$, on $\partial\Omega$, describes the velocity of heat flow through the surface. The special case when $u_G = 0$, as example, represents a perfect insulated boundary.

Mixed boundary condition. $u = u_D$ on Γ_D, $\frac{\partial u}{\partial \boldsymbol{n}} = u_N$ on Γ_N, $\partial\Omega = \Gamma_D \cup \Gamma_N$, is the most common boundary condition for cases of heat conduction in solid regions. It describes part of the domain subject to heat sources at constant temperature while another one describes the diffusion along the surface.

The heat diffusion problem is related to *initial conditions* governing the evolution in time. The initial boundary value problem can be well-posed when a solution exists and it is the unique one.

5.2 Steady Heat Transfer Boundary Value Problem

We consider a composite media described on a domain Ω with a backing part material (as example graphite) denoted with G and capsules filled with a phase change material denoted with S, such that $\Omega = G \cup S$. Let γ be the interface between S and G, Γ_D the boundary part subjected to Dirichlet boundary condition and Γ_N the Neumann one.

We observe that the two materials have different physical properties; in particular we would like to simulate a material where the conductivity term κ abruptly varies between the two parts. Considering a general heat source $g(\boldsymbol{x})$, the temperature at steady state is described so:

$$-\nabla \cdot (\kappa(\boldsymbol{x})\nabla u(\boldsymbol{x})) = g(\boldsymbol{x}), \quad \boldsymbol{x} \in \Omega = G \cup S \tag{5.10}$$

with $\kappa(\boldsymbol{x}) = (\kappa_G(\boldsymbol{x}), \kappa_S(\boldsymbol{x}))$ depending on the sub-domain considered. In our particular hybrid material, we suppose that the capsules change their phase from solid to liquid, the porosity of media along γ is such that the liquid state loses volume generating empty layers (Fig. 5.1). These layers guarantee the continuity of heat fluxes but admit a jump in temperature: considering $\boldsymbol{n}$ the normal unit vector towards γ, and R a constant representing the *contact resistance*, we resume these interface condition as

$$[\kappa(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})] = 0, \quad \text{on } \gamma \tag{5.11}$$

$$R(\kappa(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S = [u(\boldsymbol{x})], \quad \text{on } \gamma \tag{5.12}$$

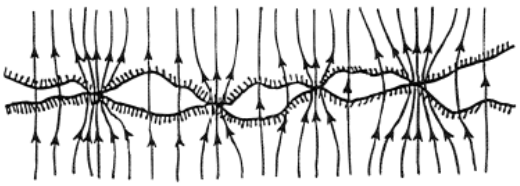


Figure 5.1: Empty layers generated by the liquid phase.

where $[\cdot]$ refers to the jump through the interface $u_G - u_S$.

We suppose that the material is contained in a tank, receiving and diffusing heat on bottom and up parts, but totally insulated on lateral boundaries. The domain Ω thus built, simplified for three salt capsules, is shown in figure 5.2.

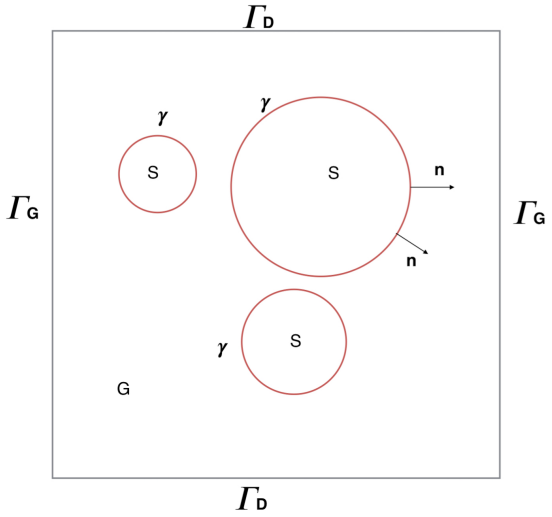


Figure 5.2: Hybrid media simplified domain

The heat conduction problem, described until this moment, considering the boundary conditions and (5.10)-(5.11)-(5.12) in a composite media with a graphite matrix foam infiltrated by a PCM is described by the following set of equations:

$$-\nabla \cdot (\kappa(\boldsymbol{x}) \bigtriangledown u(\boldsymbol{x})) = g(\boldsymbol{x}), \quad \text{in } G \cup S, \tag{5.13a}$$

$$\kappa(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}) = 0, \quad \text{on } \Gamma_N, \tag{5.13b}$$

$$u(\boldsymbol{x}) = u_D(\boldsymbol{x}), \quad \text{on } \Gamma_D, \tag{5.13c}$$

$$[k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x})] = 0, \quad \text{on } \gamma, \tag{5.13d}$$

$$R(k(\boldsymbol{x})\partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S = [u(\boldsymbol{x})], \quad \text{on } \gamma. \tag{5.13e}$$

The discontinuity along the surface caused by the jump of conductivity

κ causes a series of analytical obstacles about the existence for the solution at boundary value problem (5.13). For results about the solution, we recall the results in Chapter 1.

Nomenclature

κ	Thermal conductivity $Wm^{-1}K^{-1}$
u	Temperature $^{\circ}C$
ρ	Density $kg\,m^{-3}$
C	Heat capacity JK^{-1}
L_a	Latent heat of fusion Jg^{-1}
R	Thermal resistance m^2KW^{-1}
t	Time s

5.3 Boundary Layer Approach

In this section, we propose an analytical solution that aims to simulate the jump of temperature of the model (5.13), where we choose $g(\boldsymbol{x}) = 1$ for analytical purposes; for simplicity, the model is derived in one dimension in radial coordinates; however, two-dimensional tests of convergence are presented. We simulate the physical absence of conductivity in a fictitious layer in order to obtain a piecewise continuous solution that jumps along the layer.

As shown in Fig. 5.3 from $r = r_l = 0$, we have three values of κ, respectively κ_m, κ_a and κ_p, distributed on the subintervals $[r_l, r_m[, [r_m, r_p],]r_p, R]$ $(0 \leq r_l \leq r_m \leq r_p \leq R)$ with temperatures on the extremes $0 \leq t_l \leq t_m \leq t_p \leq t_R$.

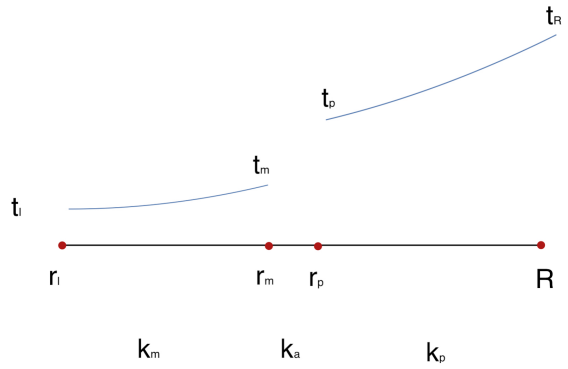


Figure 5.3: One dimensional sub-interval containing the fictitious layer.

Let t_R be the temperature on R. The κ_a part is in the interval $[r_m, r_p]$ with temperatures t_m and t_p. In what follows, we define δ such as $\delta = r_p - r_m$;

we impose the value of κ_a depending of δ:

$$\kappa_a = \alpha\delta,$$
$$r_p = r_m + \delta.$$

Let us analyse the case of left boundary condition on a subinterval; we define t_L as a temperature limit on $r_L \neq 0$ (which can be in our case r_m or r_p, namely the left boundary), κ the conductivity term, and d an additional value. We are looking for a function $\phi(r, t_L, d, \kappa, r_L)$ such that, for both subintervals $[r_m, r_p],]r_p, R]$, its derivative on r_L is equal to d and globally:

$$\partial_r(\kappa r \partial_r \phi) = r^1. \tag{5.15}$$

We look for a solution $h(r)$ of (5.15). Integrating both sides of the equation we have:

$$\partial_r h(r) = \frac{(\kappa r_L\, d + \frac{1}{2}(r^2 - r_L^2))}{\kappa r} \tag{5.16}$$
$$\Downarrow$$
$$h(r) = \int_r \frac{(\kappa r_L\, d + \frac{1}{2}(r^2 - r_L^2))}{\kappa r} = \frac{\frac{r^2}{2} + \log(r)(2d\kappa r_L - r_L^2)}{2\kappa}. \tag{5.17}$$

h_0 is its limit to r_L, and $h(r)$ derivative on r_L is equal to d:

$$h_0 = \lim_{r \to r_L} h(r) = \frac{\frac{r_L^2}{2} + \log(r_L)(2d\kappa r_L - r_L^2)}{2\kappa}. \tag{5.18}$$

Then we define:

$$\phi(r, t_L, d, \kappa, r_L) = h(r) - h_0 + t_L =$$
$$= t_L + \frac{\frac{r^2}{2} + (2d\kappa r_L - r_L^2)\log(r)}{2\kappa} - \frac{\frac{r_L^2}{2} + (2d\kappa r_L - r_L^2)\log(r_L)}{2\kappa}. \tag{5.19}$$

Where if $\cdot_L$ stands for the generic value on the left boundary condition. The function defined in (5.19) is such that its derivative on r_L is equal to d and it satisfies (5.15). For construction, however, in order to complete the steady diffusion problem internal boundary conditions have to be validated. Then we impose:

- the continuity of the solution and its fluxes at the interfaces on r_m and r_p: in order to satisfy (5.13d);
- the temperature on R equal to t_R.

[1] We remind that $r\partial_r = x\partial_x + y\partial_y$

We build a function $\omega(r) : [0, r_m] \to \mathbb{R}$ such that its derivative on 0 is zero value, $\omega(0) = c$ and it satisfies (5.15):

$$\omega(r) = c + \frac{r^2}{4\kappa_m}.$$

The unknowns are c (temperature on 0), d_m the right derivative on r_m, t_m the temperature on r_m, t_p the temperature on r_p and d_p the right derivative on r_p. We have thus five equations for five unknowns:

$$\begin{aligned}
\kappa_m \partial_r \omega(r_m) =& \kappa_a d_m, \\
\omega(r_m) =& t_m, \\
\kappa_a \partial_r \phi(r_p, c, d_m, \kappa_a, r_m) =& \kappa_p d_p, \\
\phi(r_p, t_m, d_m, \kappa_a, r_m) =& t_p, \\
\phi(R, t_p, d_p, \kappa_p, r_p) =& t_R.
\end{aligned}$$

This system is solved with the solutions:

$$c = -\frac{\kappa_a\kappa_m R^2 + \kappa_a\kappa_p r_m^2 - \kappa_m\kappa_p r_m^2 - \kappa_a\kappa_m r_p^2 + \kappa_m\kappa_p r_p^2 - 4\kappa_m\kappa_a\kappa_p t_r}{4\kappa_m\kappa_a\kappa_p}, \tag{5.20a}$$

$$t_m = -\frac{\kappa_a R^2 - \kappa_p r_m^2 - \kappa_a r_p^2 + \kappa_p r_p^2 - 4\kappa_a\kappa_p t_r}{4\kappa_a\kappa_p}, \tag{5.20b}$$

$$d_m = \frac{r_m}{2\kappa_a}, \tag{5.20c}$$

$$t_p = -\frac{R^2 - r_p^2 - 4\kappa_p t_r}{4\kappa_p}, \tag{5.20d}$$

$$d_p = \frac{r_p}{2\kappa_p}. \tag{5.20e}$$

We evaluate the difference J between the temperatures and the fluxes F_1, F_2 on the interfaces:

$$\begin{aligned}
J = t_p - t_m &= \frac{r_p^2 - r_m^2}{4\kappa_a}, \\
F_1 = \kappa_a d_m &= \frac{r_m}{2}, \\
F_2 = \kappa_p d_p &= \frac{r_p}{2}.
\end{aligned}$$

We remind the definition of a real value δ in (5.14). From a study on limit to zero of δ, calculating the jump of temperature in function of this limit, we obtain:

$$\lim_{\delta\to 0} J = \lim_{\delta\to 0} \frac{(r_m + \delta)^2 - r_m^2}{4\alpha\delta} = \lim_{\delta\to 0} \frac{r_m + \delta}{\alpha} = \frac{r_m}{\alpha}.$$

A calculus at limits, taking $\alpha = \frac{2}{\beta}$ (eq. (5.13e)), arises a proportionality between the jump J and the flux F_1, such that:

$$\beta F_1 = \frac{r_m}{\alpha} = \lim_{\delta \to 0} J$$

This calculus of limit complete the proofs for the hypothesis proposed in eq. (5.13), for this kind of modelisation with a fictitious boundary layer support. Resuming, on different subintervals and their relative variables using (5.20), the function that satisfies the constraints is:

- $r \in [0, r_m[$:
  $\omega(r) = c + \frac{r^2}{4\kappa_m} =$
  $$-\frac{\kappa_a\kappa_m R^2 + \kappa_a\kappa_p r_m^2 - \kappa_m\kappa_p r_m^2 - \kappa_a\kappa_m r_p^2 + \kappa_m\kappa_p r_p^2 - 4\kappa_m\kappa_a\kappa_p t_r}{4\kappa_m\kappa_a\kappa_p} + \frac{r^2}{4\kappa_m};$$
- $r \in [r_m, r_p]$:
  $\phi(r, t_m, d_m, \kappa_a, r_m) = \phi(r, -\frac{\kappa_a R^2 - \kappa_p r_m^2 - \kappa_a r_p^2 + \kappa_p r_p^2 - 4\kappa_a\kappa_p t_r}{4\kappa_a\kappa_p}, \frac{r_m}{2\kappa_a}, \kappa_a, r_m);$
- $r \in]r_p, R]$:
  $\phi(r, t_p, d_p, \kappa_p, r_p) = \phi(r, -\frac{R^2 - r_p^2 - 4\kappa_p t_r}{4\kappa_p}, \frac{r_p}{2\kappa_p}, \kappa_p, r_p).$

5.3.1 Numerical Results

The model proposed above provides a new subdomain where the conductivity term $\kappa(\vec{x})$ tends to 0 $Wm^{-1}K^{-1}$ to guarantee a discontinuity on the internal part. An example is given in Figure 5.4), where the values κ_m and κ_p are, respectively, 1 and 10 $Wm^{-1}K^{-1}$. The discontinuity is described as a double mollified function (similar to the one proposed in 4.3):

$$\kappa(\boldsymbol{x}) = \alpha + (\beta - \alpha)\left(\frac{\tanh\left(\sigma_1 \cdot \Phi_1(\boldsymbol{x})\right) + 1}{2}\right) + (\psi - \beta)\left(\frac{\tanh\left(\sigma_2 \cdot \Phi_2(\boldsymbol{x})\right) + 1}{2}\right), \tag{5.21}$$

where $\Phi(x)_{1,2}$ are the distance functions with respect to the interfaces of discontinuity at radius r_m and r_p respectively, and $\sigma_{1,2}$ are the regularisation parameters.

5.3.2 The Mesh Refinement Strategy

As seen in Section 4.3, using an adaptive mesh refinement approach is the best strategy to focus the mesh along the mollification parts with a larger layer to cover the finite difference stencil. On the other hand, the model above proposed that we reconstruct on the squared domain $[0, 1] \times [0, 1]$ converges with $\delta \to 0$. As a result of these observations the mesh refinement follows the criteria:

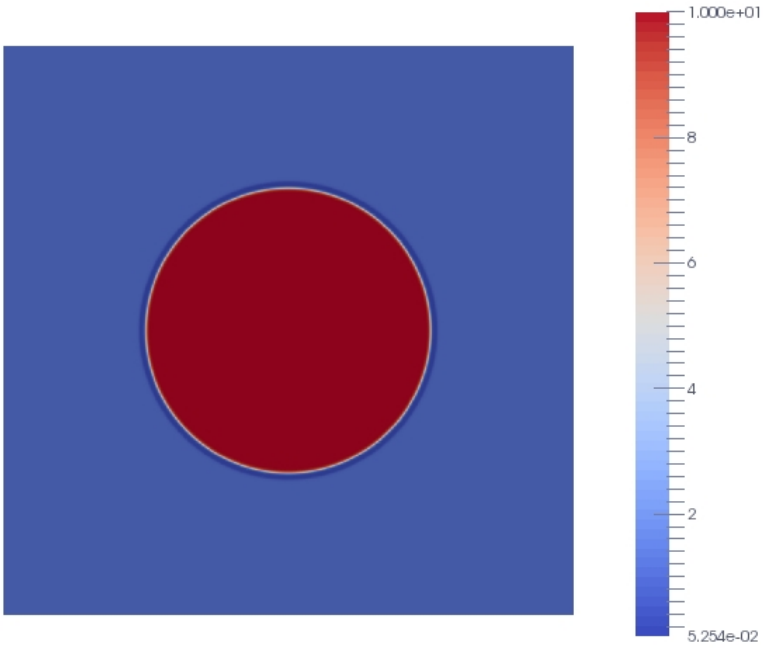


Figure 5.4: Double mollified function $\kappa(\vec{x})$ from 1 to 10.

- the maximal depth of the tree is fixed from $M = 9$ to $M = 14$;
- the δ-layer zone is defined proportionally to M;
- the squared domain is initially uniformly meshed with level $M - 3$;
- from $M - 3$ to M, the mollified function (5.21) is evaluated on each octant. If the octant lies on the regularization zone[2] it splits into four children;
- balance constraints are applied on the jump zones.

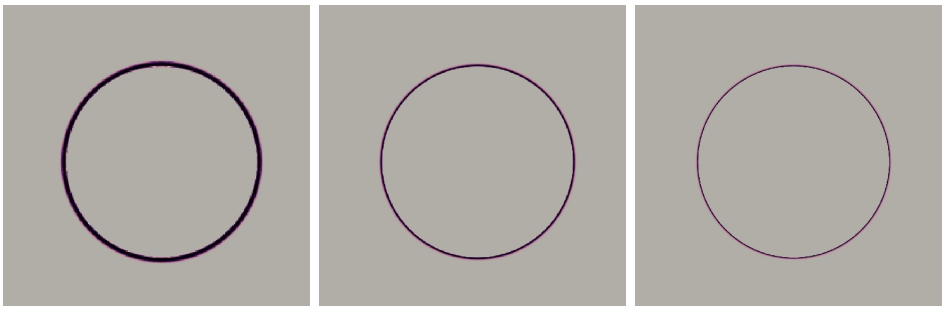
The regularization of the layer zone in order with the level of the tree allows us to obtain a simulation of the limit of $\delta \to 0$ with the fictitious layer, which decreases with $O(\Delta x)$ as shown in figure 5.5.

The convergence results for this test case are given in 5.1

5.4 The Heat Conduction Problem With Phase Change In A Composite Media

By *enthalpy*, we mean measurement of energy in a thermodynamic system. It is the thermodynamic quantity equivalent to the total heat content of a

[2]The regularization zone is defined by $|\nabla\kappa(\boldsymbol{x})| > \epsilon$, where ϵ is a small user defined parameter.



(a) Level 9 (b) Level 10 (c) Level 11

Figure 5.5: Boundary layer zone behaviour in order with the level of refinement.

Δx	Norm Infty	Norm 2	Mesh Points
$1.953 \cdot 10^{-3}$	$1.96 \cdot 10^{0}$	$1.024 \cdot 10^{-2}$	20500
$9.766 \cdot 10^{-4}$	$9.885 \cdot 10^{-1}$	$5.108 \cdot 10^{-3}$	64180
$4.883 \cdot 10^{-4}$	$4.913 \cdot 10^{-1}$	$2.524 \cdot 10^{-3}$	211324
$2.441 \cdot 10^{-4}$	$2.449 \cdot 10^{-1}$	$1.254 \cdot 10^{-3}$	773056
$1.22 \cdot 10^{-4}$	$1.224 \cdot 10^{-1}$	$6.256 \cdot 10^{-4}$	2959756
$6.103 \cdot 10^{-5}$	$6.131 \cdot 10^{-2}$	$3.131 \cdot 10^{-4}$	11167768

Table 5.1: Convergence results with $\delta = \Delta x(\frac{2}{2^{M-1}})$.

system and includes the internal energy plus the product of pressure and volume. Enthalpy is defined as a state function that depends only on the prevailing equilibrium state identified by the system's internal energy, pressure, and volume.

We assume that the internal salt capsules change their phase from solid to liquid at constant temperature u_f. We also assume the specific heat and the thermal conductivities are phase-wise constant. The idea of the enthalpy approach is based on the fact that the energy conservation law, expressed in terms of enthalpy and temperature, together with the equation of state contain all the physical information needed to determine the evolution of the phases.

We consider $-\boldsymbol{q}\cdot\boldsymbol{n}$ to be the heat flux into the volume Ω. With constant pressure, we consider a new function, $H(u)$, that varies in volume with the fluxes along the surface, such that:

$$\partial_t \int_\Omega H \, d\Omega = \int_{\partial\Omega} -\boldsymbol{q}\cdot\boldsymbol{n} \, \partial\Omega.$$

Given the relationship of the enthalpy with the heat flux, if the total variation of enthalpy $\Delta H > 0$, the heat processes is said to be *endothermic*; in the opposite case, it is *exothermic* (Fig. 5.6).

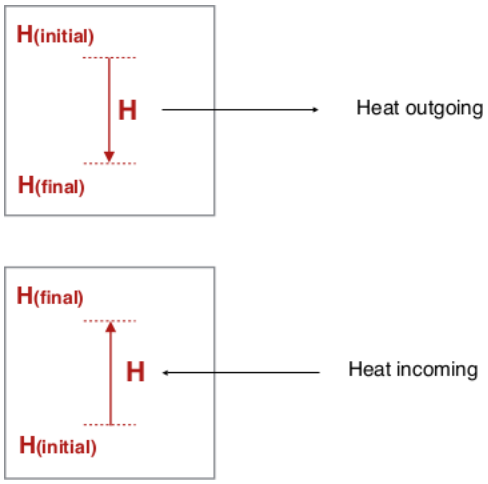


Figure 5.6: Enthalpy behaviour in order with the heat flux.

We recall the law (5.1), which allows us to rewrite the relationship above as:

$$\partial_t \int_\Omega H \, d\Omega = \int_{\partial\Omega} \kappa \nabla u \cdot \boldsymbol{n} \, \partial\Omega$$

$$\Downarrow$$

(Using the divergence theorem)

$$\partial_t H - \nabla \cdot (\kappa \nabla u) = 0, \tag{5.22}$$

obtaining the general heat conduction equation. The enthalpy $H(u)$ can be defined as a function of the latent heat

$$H(u^*) = \int_0^{u^*} \rho C \, du + \rho f(u) L_a, \tag{5.23}$$

where L_a is the latent heat of fusion, $f(u)$ is a function that describes the liquid fraction and C is the specific heat.

Adding the new hypothesis to the diffusion problem (5.13), we can resume the heat diffusion problem in a composite media, with internal heat sources, internal interfaces with discontinuities, and boundary conditions, using the following equations:

$$\begin{aligned}
\partial_t H(u(\boldsymbol{x})) - \nabla \cdot (\kappa(\boldsymbol{x}) \bigtriangledown u(\boldsymbol{x})) &= g(\boldsymbol{x}), && \text{in } G \cup S, && \text{(5.24a)}\\
\kappa(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, && \text{on } \Gamma_N, && \text{(5.24b)}\\
u(\boldsymbol{x}) &= u_D(\boldsymbol{x}), && \text{on } \Gamma_D, && \text{(5.24c)}\\
[k(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x})] &= 0, && \text{on } \gamma, && \text{(5.24d)}\\
R(k(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S &= [u(\boldsymbol{x})], && \text{on } \gamma. && \text{(5.24e)}
\end{aligned}$$

5.4.1 Enthalpy Formulation

Enthalpy exhibits a jump at the points where $u(\boldsymbol{x}) = u_f$, which in some cases may provoke numerical instabilities. In order to overcome these difficulties often linearised expressions of the enthalpy and the liquid fraction are used.

Denoting the variables with G, if concerning the host material, with S if concerning the phase change material, and with s and l if differences between solid and liquid states exist, we suppose that:

- the specific heat C is phase-wise constant with values C_G, C_S^s, C_S^l;
- the density ρ is not affected by the phase change with values ρ_G, ρ_S;
- the conductivity κ is a piecewise continuous function or constant with values κ_G, κ_S.

In the field of phase changing materials V.R. Voller [67] proposes an overview on the solutions of problems concerning nonlinear phenomena. His analysis involves the tracking of a sharp during the change boundary due to phase change. The enthalpy formulation used by Voller is taken up by F. Jelassi, M. Azaïez and E. Palomo Del Barrio in [68], in which they define the volumetric enthalpy function and the liquid fraction as follows:

$$H(u) = \begin{cases} [(\rho C)_S^s(1 - f(u)) + (\rho C)_S^l f(u)]u + L_a f(u), & \text{in } S, \\ (\rho C)_G u, & \text{in } G. \end{cases} \tag{5.25}$$

$$f(u) = \frac{1}{2} + \frac{1}{2}\tanh(\sigma(u - u_f)), \tag{5.26}$$

with σ a regularization parameter.

In the numerical test presented in this work we use a linearised formulation of enthalpy such that:

$$H(u) = (\rho C)_G u, \quad \text{in } G, \tag{5.27}$$

on S depending on the melting temperature u_f and a user-defined parameter ϵ

$$H(u) = \begin{cases} (\rho C)_S^s u & \text{if } u < u_f - \epsilon, \\ (\rho C)_S^s u + \rho_S^l L_a f(u) & \text{if } u_f - \epsilon < u < u_f + \epsilon, \\ (\rho C)_S^l u + \rho_S^l L_a & \text{if } u_f + \epsilon < u. \end{cases} \tag{5.28}$$

$$f(u) = \begin{cases} 0 & \text{if } u < u_f - \epsilon, \\ \frac{u - u_f + \epsilon}{2\epsilon} & \text{if } u_f - \epsilon < u < u_f + \epsilon, \\ 1 & \text{if } u_f + \epsilon < u. \end{cases} \tag{5.29}$$

The formulations (5.28)-(5.29) are inspired by X. Jin et al. [69], where a central difference scheme is applied in space and a fully implicit scheme in

time. They used this method to compare two of the most common different heat transfer models for phase change materials: the effective heat capacity method and the enthalpy method. They conclude that for the heat capacity method a significant error is obtained for some attribution in controlling the PCM fusion temperature; on the other hand, the enthalpy model needed more computing time. A fast implicit finite difference method for the analysis of phase changing problems is given by Voller in [70]- [71], where the *new source* method is presented. This method consists in adding a source temperature for the heat equation; this tool would allow to compute by iterative steps the latent heat appearing during the phase changing at melting temperature.

5.4.2 Pseudo-Time Scheme

To solve this kind of problem, the classical methods to solve PDEs are often combined to approaches that allow us to neglect the internal discontinuities. In [68] finite element substructuring algorithm is presented for steady and unsteady problems. The authors used the interface Steklov–Poincare operator and applied an iterative sub-structuring method. This way of describing and solving the problem is well adapted to the specificities of the studied composites, which include thermal contact resistances at the interface, giving rise to temperature jumps at these locations. An analysis of the latent heat and its properties during the liquid phase is given by [72], which offers various combinations of conductivity values for the hybrid material mentioned, studying the interaction of graphite foam in the phase-change material and the turbulent flow.

To discretize problem (5.24), we introduce a fictitious fixed time step in the time-marching scheme obtaining an implicit scheme. Considering (5.24a) with $g(\boldsymbol{x}) = 0$ at time $t = 0s$ (absence of internal heat sources when the system is turned off at the beginning of computation), we rewrite the expression for a general time step t_n, with $\Delta t = t_{n+1} - t_n$, and with u_n the temperature at time t_n.

$$\frac{H(u_{n+1}) - H(u_n)}{\Delta t} - \nabla \cdot (\kappa \nabla(u_{n+1})) = 0, \tag{5.30}$$

where for simplicity we denote with κ the piecewise continuous $\kappa(\boldsymbol{x}) = (\kappa_G(\boldsymbol{x}), \kappa_S(\boldsymbol{x}))$. Let be $\varepsilon = \frac{1}{\Delta t}$:

$$(5.30) \Rightarrow \varepsilon H(u_{n+1}) - \nabla \cdot (\kappa \nabla(u_{n+1})) = \varepsilon H(u_n). \tag{5.31}$$

We introduce a new variable T, fictitious temperature, adding a pseudo-time term $\frac{\partial T}{\partial \tau}$ to equation (5.31), and we assume:

- $T_n = u_n$ for the generic time step t_n;
- $T_0 = u_n$ as iterative initial condition.

We obtain an internal loop from the solution at time t_n to the t_{n+1} one, such that

$$\frac{T_{j+1}-T_j}{\Delta\tau}+\varepsilon H(T_j)-\nabla\cdot(\kappa\nabla(T_{j+1}))=\varepsilon H(T_n),\ \forall j\in\{0,\dots,J_{max}\}$$

$$\Downarrow$$

$$\frac{T_{j+1}}{\Delta\tau}+-\nabla\cdot(\kappa\nabla(T_{j+1}))=\varepsilon H(T_n)+\frac{T_j}{\Delta\tau}-\varepsilon H(T_j),\ \forall j\in\{0,\dots,J_{max}\}. \tag{5.32}$$

The loop on j is such that:

- from 0 to J_{max} we are able to compute $\varepsilon H(T_n)+\frac{T_j}{\Delta\tau}-\varepsilon H(T_j)$ since T_n and T_0 are given by hypothesis and $H(\cdot)$ is obtained from (5.28)-(5.29);
- we can compute the discretization matrix $\frac{T_{j+1}}{\Delta\tau}+-\nabla\cdot(\kappa\nabla(T_{j+1})$ easily with our method using the tools seen above with an additive term along diagonal elements.

5.4.3 Convergence Versus The Steady State

We have not specified any value, until now, for J_{max}; if the solution is smooth enough J_{max} can be fixed as small as needed for each relaxed time step, obtaining a convergent approach, but since the temperature is not known a priori, we applied a variable value J_{max}, determined during the computation.

The formulation (5.32) tends to (5.30) as long as

$$\frac{T_{j+1}-T_j}{\Delta\tau}\to 0.$$

Thus, we determine $j+1=J_{max}$ if and only if

$$||T_{j+1}-T_j||_\infty<\epsilon_1.^3 \tag{5.33}$$

When (5.33) is satisfied, we impose the new solution at time step t_{n+1}

$$u_{n+1}=T_{J_{max}}, \tag{5.34}$$

that leads back to equation (5.30). At the same time, the temperature in time converges to steady state, equilibrium of heat internal and external sources for the composite media, as long as

$$\frac{H(u_{n+1})-H(u_n))}{\Delta t}\to 0.$$

Thus, when a new solution u_{n+1} is computed two cases are possible:

- the new solution satisfies the convergence condition

$$\frac{H(u_{n+1})-H(u_n)}{\Delta t}-\nabla\cdot(\kappa\nabla(u_{n+1}))<\epsilon_2{}^4, \tag{5.35}$$

[3] ϵ_1 is an user defined tolerance.

[4] ϵ_2 is an user defined tolerance.

which approximates (5.30) as good as possible in calculus at a limit for $\epsilon_2 \to 0$;

- the condition (5.35) is not satisfied, the iterative procedure is repeated for the new initial condition $T_0 = u_{n+1}$, reaching the solution for the subsequent time step until convergence.

5.4.4 Numerical Results

Our numerical simulations provides different number of molten salt capsules where we aim to solve the problem (5.24). We described the enthalpy during the computation using the formulations (5.28)-(5.29)

Numerical Domain Conditions

The presented method is applied to solve the heat diffusion problem on composite materials. We propose a boundary part of the domain insulated, the other part with a heat source, and variable positions and properties for salt capsules. We solve the problem on the logic unit square $\Omega = [0,1] \times [0,1]$ as the container of our composite material (see figure 5.7).

$$\begin{aligned}
\partial_t H(u(\boldsymbol{x})) - \nabla \cdot (\kappa(\boldsymbol{x}) \bigtriangledown u(\boldsymbol{x})) &= 0, && \text{in } G \cup S,\\
\kappa(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x}) &= 0, && \text{on } \Gamma_N,\\
u(\boldsymbol{x}) &= u_D, && \text{on } \Gamma_D,\\
[k(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x})] &= 0, && \text{on } \gamma,\\
R(k(\boldsymbol{x}) \partial_{\boldsymbol{n}} u(\boldsymbol{x}))_S &= [u(\boldsymbol{x})], && \text{on } \gamma.
\end{aligned}$$

Diffusion Test Case

We applied the pseudo-time scheme on a case inspired by tests presented in [7] and [73]. In this first test, we suppose three encapsulated parts. The three capsules are similar but not of the same dimensions; however, they have common physical properties (conductivity, latent heat, density,...).

We impose the initial temperature of the system $u_0 = -10°C$, the melting temperature of the PCM material at $u_f = 0°C$, a part of $\partial\Omega$ subject to Dirichlet boundary condition Γ_D as a continuous constant heat source of $u_D = 20°C$. Meanwhile, the other part of boundary Γ_N is totally insulated with Neumann boundary conditions. Moreover, in this test, the resistance term for the material tends to zero from $\sim 10^{-5}$ to $\sim 10^{-9}$ $m^2 K W^{-1}$; for simplicity, we imposed this term R at zero; also, tests with different values of R were done but not presented because of the low influence of its variation.

We tested different values of the conductivity κ, supposing for the conductivity at liquid state equals the solid one, using the mollified formulation (4.6) (Fig.5.8b). The first and simplest case proposed presents $L_a = 2.1 J g^{-1}$, $C_G = 10 J K^{-1}$ $C_S^s = C_S^l = 21.5\ J K^{-1}$.

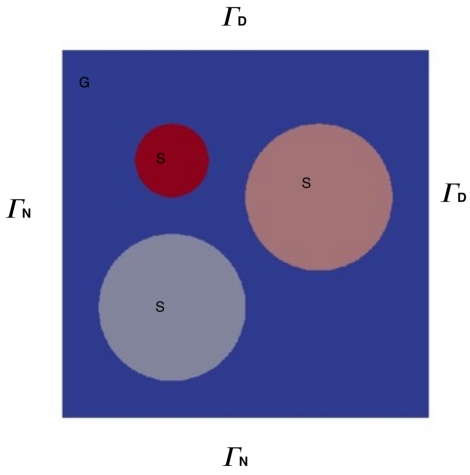


Figure 5.7: Numerical domain, host media containing three PCM capsules.

The time step considered is $\Delta t = 0.1s$; the pseudo-time between two solutions is $\Delta\tau = 0.01$.

We impose the temperature, needed to compare the error progress, computed as the inverse function of enthalpy. We remember that the latent heat of the host material is zero-value:

$$T = \begin{cases} \frac{H}{\rho C} & \text{if } u < u_f - \epsilon, \\ \frac{H - \rho L_a \frac{\epsilon - u_f}{2\epsilon}}{\rho C + \frac{\rho L_a}{2\epsilon}} & \text{if } u_f - \epsilon < u < u_f + \epsilon, \\ \frac{H - \rho L_a}{\rho C} & \text{if } u_f + \epsilon < u. \end{cases} \tag{5.36}$$

For the presented test we refined the mesh until level 9 of the tree using an AMR approach to focus the salt parts; meanwhile, the host part is at level of refinement 6, obtaining in total 133306 degrees of freedom; balancing constraints are applied to control the jump (see Fig. 5.8a).

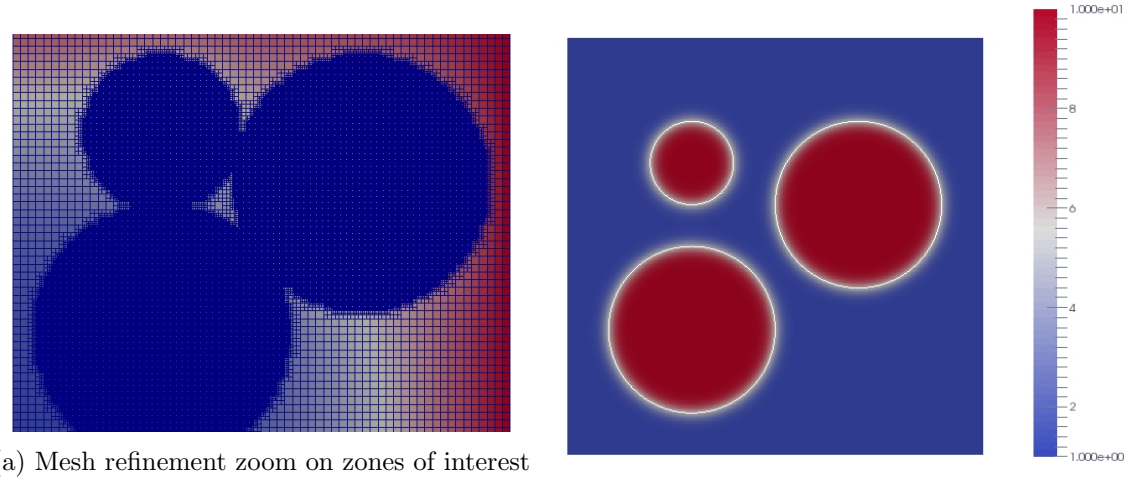
Pseudo-Time Convergence

With respect to condition

$$\frac{T_{j+1} - T_j}{\Delta\tau} \to 0,$$

a study of convergence for a pseudo-time processes is proposed in order to analyse its convergence. The case reported in figure 5.9 refers to the amount of time, from 2.1 seconds to 2.2 seconds, taking in example the last 10 iterations.

In figures 5.10, the front propagation of temperature at different time steps is presented; for the last one, we reported the liquid fraction computation, which highlights the front of fusion temperature u_f for two liquefied spheres and the last one still in solid state.



(a) Mesh refinement zoom on zones of interest

(b) Example test $(\kappa_G, \kappa_S) = (1, 10)$

Figure 5.8: Refinement of computational domain zoom (left) and conductivity term (right).

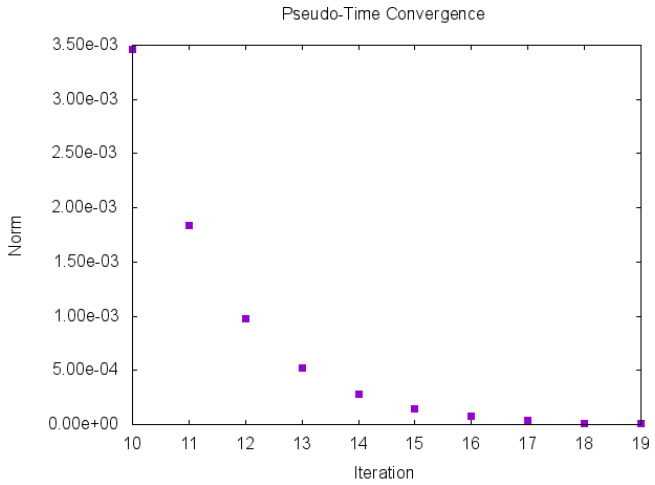
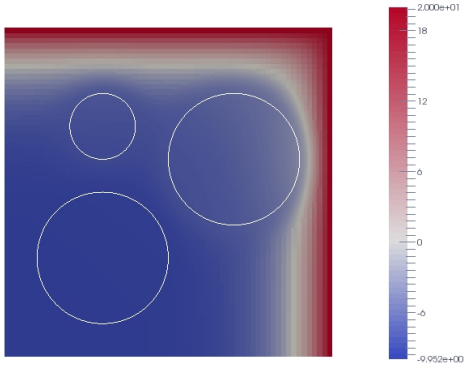


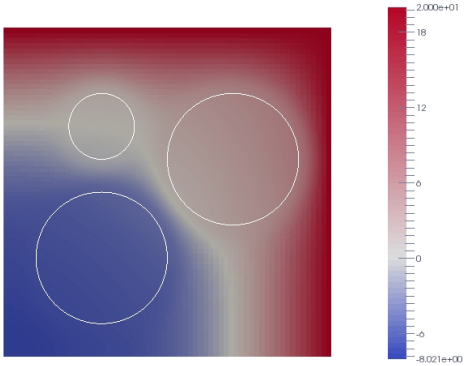
Figure 5.9: Rate of convergence of internal pseudo-time relaxation. From $t_n = 21\Delta t$ to $t_{n+1} = 22\Delta t$. Convergence reached at $J_{max} = 19$.

We talked about the fact that the conductivity of the material, in the face of its high capacity, is low compared to that of the conductive material that supports it. We inversed the problem, retaining the conditions proposed with $(\kappa_G, \kappa_S) = (100, 1)\ Wm^{-1}K^{-1}$ until reaching convergence. We imposed $\varepsilon_2 = 10^{-6}$.

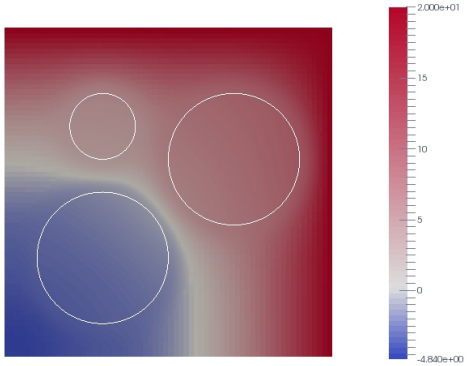
The time step considered is $\Delta t = 0.07s$; the pseudo-time between two solutions is $\Delta\tau = 0.01$. The convergence is obtained approximately at $t_n \sim 30\Delta t$ because of the good conductivity of graphite foam. We will see,



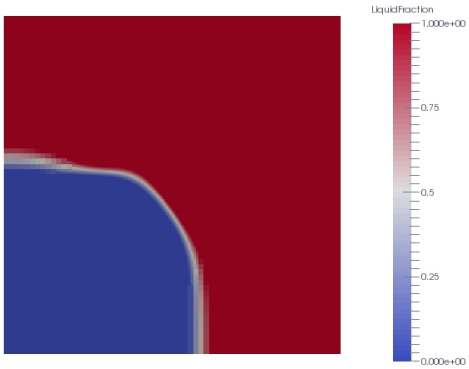
(a) Temperature at $t_n = \Delta t$



(b) Temperature at $t_n = 5\Delta t$



(c) Temperature at $t_n = 8\Delta t$



(d) Liquid fraction evaluation at $t_n = 8\Delta t$

Figure 5.10: $(\kappa_G, \kappa_S) = (100, 1)\ Wm^{-1}K^{-1}$. Propagation front of temperature u_f in time and liquid fraction (right bottom).

in what follows, parameters that better describe the real physics of the materials.

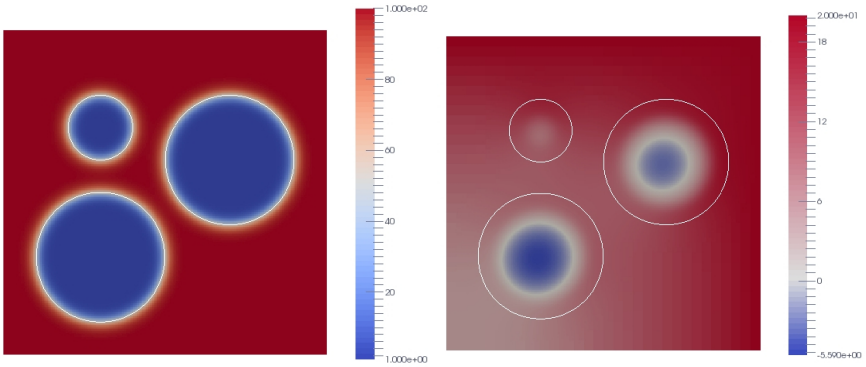


Figure 5.11: $(\kappa_G, \kappa_S) = (100, 1)$. Temperature at $t_n = 2\Delta t$

Charge Time Test

The main idea of this test is to measure and compare the charge time of a system in the presence of molten salt. We suppose we to have three salt capsules aligned in square domain $\Omega = [0,1] \times [0,1]$, as in Figure 5.12, with left and right walls insulated and heat coming from bottom to upper side. We define *time of charge* of this system as the time that occurs from the initial temperature to the complete fusion of salt capsules; the temperature u is fixed on boundaries as $u_D = 400°C$ at the lower wall and $u_D = 300°C$ at the upper one.

The composite material physical properties are given in [74], which studied the numerical effect of an increasing resistance for these kinds of problems. Other details are given in [75]. We briefly resume the parameters they computed numerically: the fusion temperature of PCM is $u_f = 306°C$, the conductivity term has values $(\kappa_G, \kappa_S) = (40, 0.514)\ Wm^{-1}K^{-1}$, the latent heat of fusion is set at 187 Jg^{-1}, and the density $(\rho_G, \rho_S) = (240, 1900)$ $kg\,m^{-3}$.

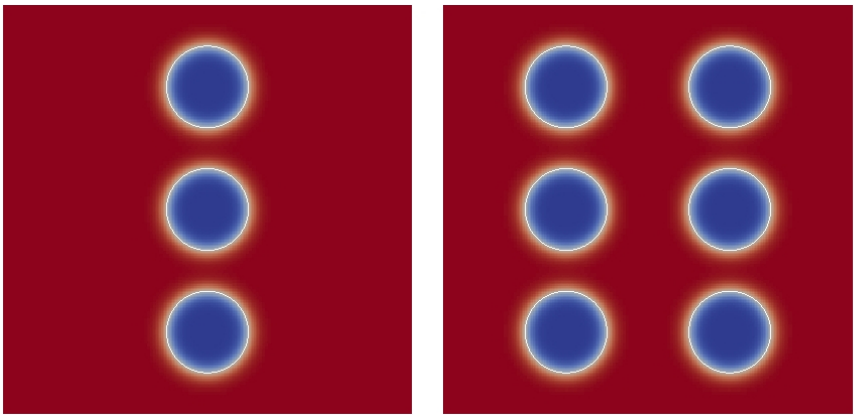


Figure 5.12: Conductivity term $(\kappa_G, \kappa_S) = (40, 0.514)\ Wm^{-1}K^{-1}$.

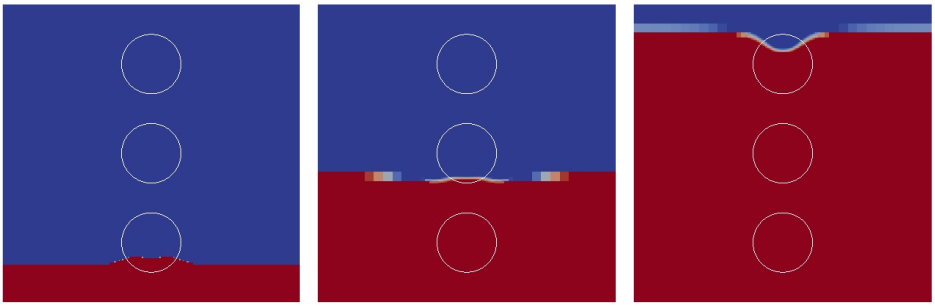


Figure 5.13: Liquid fraction propagation front: three capsules.

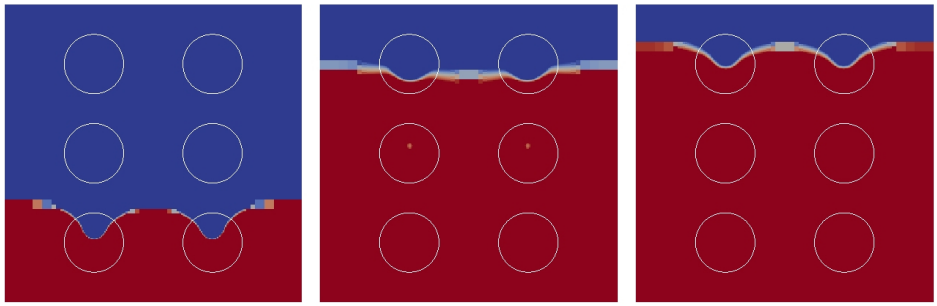


Figure 5.14: Liquid fraction propagation front: six capsules.

A first test compared the charge time for a system composed of three capsules of radius 0.1 aligned in the centre of the domain. The second one is composed of six capsules aligned in two columns of three PCM sections of radius 0.1. The time step is coherent with the physics of materials imposed at $\Delta t = 200s$ for both cases.

In Figures 5.13 and 5.14, the propagation liquid fraction along the numerical domain is presented. Similarly, the temperature behaviour is presented in Figures 5.15 and5.16, where the melting temperature u_f is used as an intermediate value (white-coloured) to describe the *cooler* zones at best.

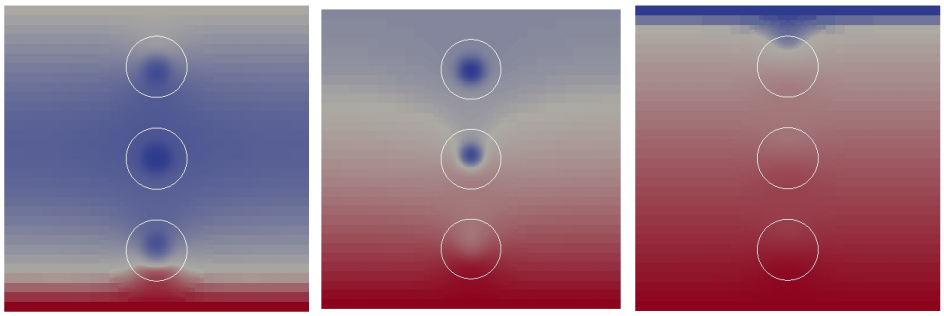


Figure 5.15: u_f propagation front for three capsules case.

We compared the times of these simulations. We expected that increasing the surface of salt capsules would correspond to an increase in convergence time; in fact, the low conductivity of salt creates a clear slowdown on the heat temperature propagation. We computed a convergence time for the three capsules case of $t_n \sim 120\Delta t \sim$ less than 7 hours and $t_n \sim 150\Delta t \sim$ more than 8 hours for the second one. The comparison of temperatures respectively after $4200sec$ and $6000sec$ is given in Figures 5.17 and 5.18. The comparisons highlight a more rapid propagation of temperature where the surface of graphite is prevalent.

We remark that we did not compare the code times because, following the salt surfaces, we modified the degrees of freedom between the two cases;

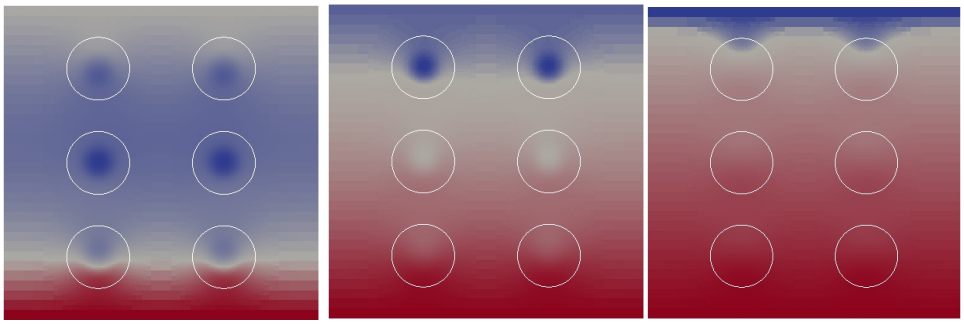


Figure 5.16: u_f propagation front for six capsules case.

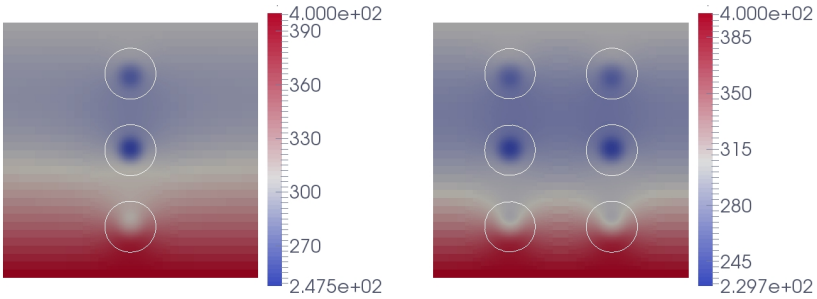


Figure 5.17: u_f propagation front comparison. $t_n = 21\Delta t$.

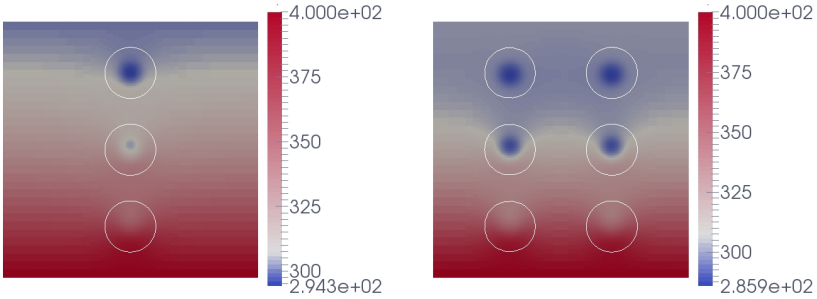


Figure 5.18: u_f propagation front comparison. $t_n = 30\Delta t$.

however, the simulations ran on 48 cores, reaching convergence in less than four hours for both cases.

Once we determine an increase in time proportional to salt surface in symmetric and regular cases, we can change the geometry with the more complex presence of salt capsule. The last result for heat diffusion in this composite material is given using ten capsules with different dimensions and irregular position. In Figure 5.19 the conductivity term is presented. We retain the physical properties referred above. The liquid fraction front propagation at different times is shown in Figure 5.20; moreover, in Figure 5.21, the propagation of melting temperature ($u_f = 306°C$) is given.

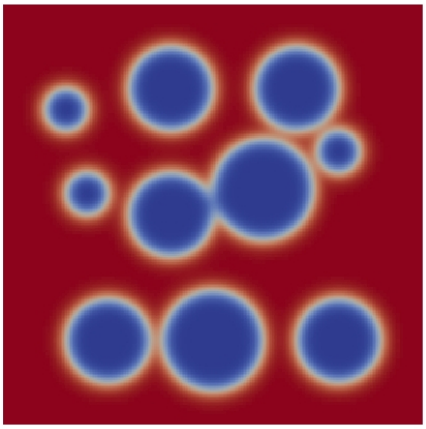


Figure 5.19: Conductivity term $(\kappa_G, \kappa_S) = (40, 0.514)\ Wm^{-1}K^{-1}$.

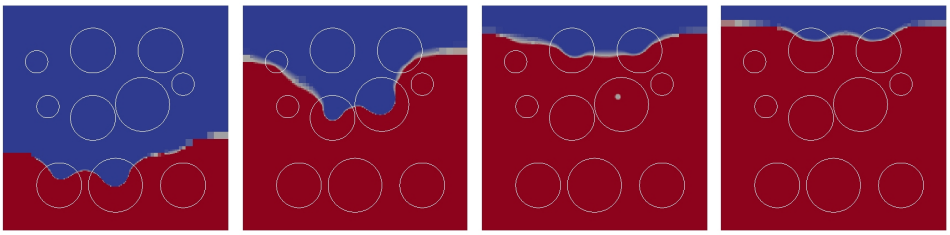


Figure 5.20: Liquid fraction propagation front: ten sparse capsules.

This test makes the increase in convergence times more apparent; in fact, the material reaches steady state stability after more than 11 hours $\sim$, overcoming $t_n = 200\Delta t$; meanwhile, the code's execution time, as above on 48 processors, remains under two ours.

5.5 Preliminary Conclusion

In this chapter, we presented an analysis of the heat equation, the physical properties that characterize the heat diffusion in general, and mixed materials in particular. We presented a boundary layer approach able to impose the jump through internal boundaries. We proved the convergence in order with the size of the mesh.

We further explored the problem of heat conduction in mixed materials; in this part we presented the enthalpy term in detail, its analytical formulation and the linearised one computed in our code. To solve the problem in time, we devised an implicit scheme, and we served a pseudo-time relaxation. Thus, we obtained simulations in time for heat conduction problems on composite media with a phase-changing material part encapsulated in graphite foam. The presented results properly follow the physics expected,

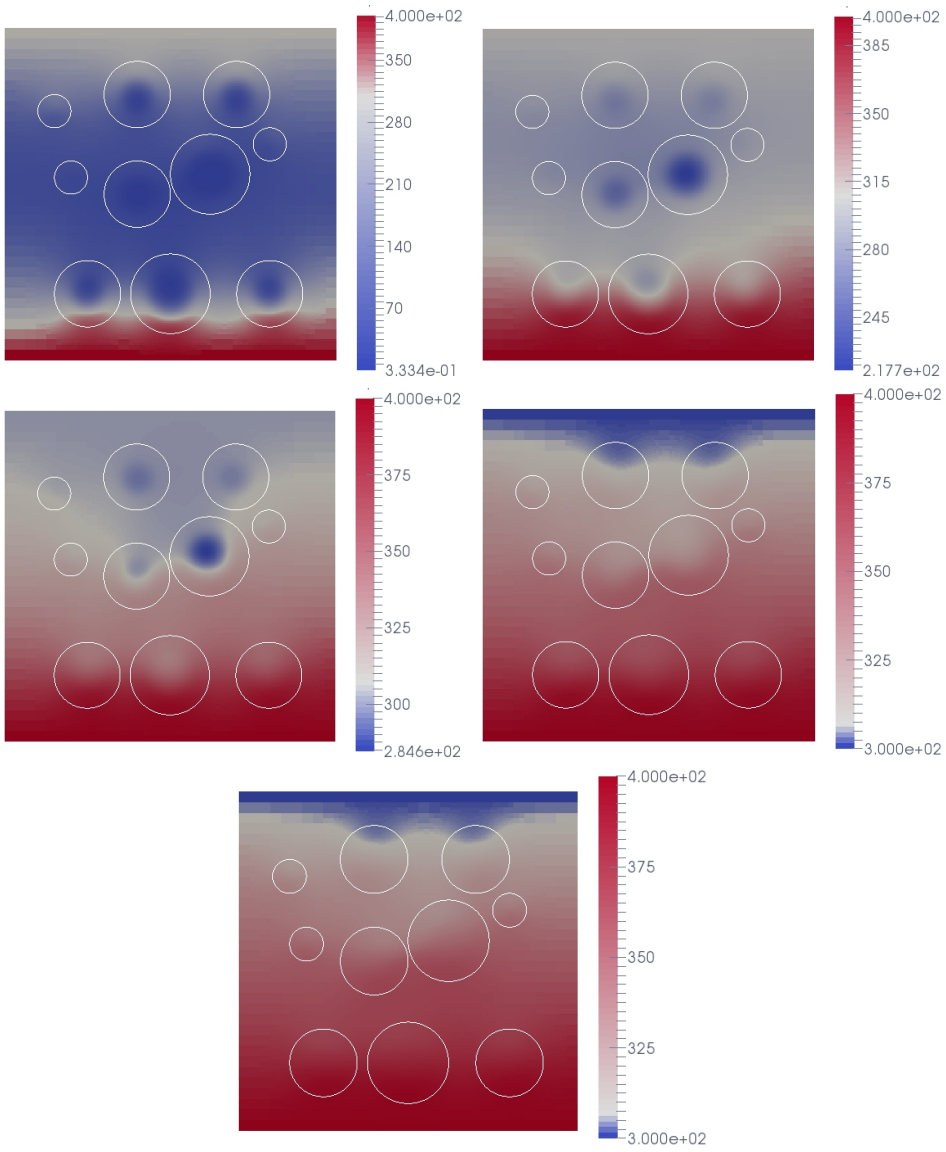


Figure 5.21: u_f propagation front for ten sparse capsules case.

and we were able to prove the time-increasing dependence of the percentage of both materials.

A future perspective is to extend the results in three-dimensions, adding the term of resistance between the sub-domains. Moreover, further analysis of dependency between convergence times and salt volumes are predicted.

Conclusion

In this thesis we have proposed a numerical method to solve the elliptic equation with mixed boundary conditions and discontinuous coefficients. The problem is discretized on a hierarchical Cartesian quadtree-based grid by a finite-difference method. The values on jumps are computed using a minimization problem to extrapolate the finite-difference weights to ensure consistency and tend to second-order convergence where possible. This computation guarantees the supra-convergence of the method globally. Also most tests proved that where second order is established on the major part of the mesh, globally second-order accuracy can be obtained. This is the main strength and drawback of the discretization. It is a strong point because the method is easy to obtain, conceptually simple and coherent with the real topology of concerned problems. On the other hand, it generates difficulties for the obtained matrix that describes the discretization, which is sparse and difficult to invert; this drawback requires special attention to the choice on solvers in parallel.

First, the model problem was presented in detail. The many applications that involve Poisson's equations are associated with studies in the fields of mechanics, physics of conducting media, electrostatics and gravitational potential. The application proposed in this thesis is a heat-conduction problem, and in particular the category of variable coefficient Poisson equation where coefficients abruptly change through different geometries included in the resolution domain. The primary objective was to evidence the difficulties that affect a Poisson problem with internal discontinuities and variable coefficients; then we also reported a proof of existence for weak solutions of our specific problem. We presented the strategies to overcome the obstacles mentioned and possible alternative spatial discretizations that, let us remember, strongly affect the total cost of numerical calculation. To face complex geometrical problems, standard discretization methods require multilevel solutions that can result in being hard to compute; moreover, the geometry might evolve in time, which would require a new discretization at each time step. We listed different methods to define the vast category of resolutions among which the current work is placed.

At the centre of each numerical method for solving differential equations

is the way in which we discretize a continuous domain of interest in a grid of many individual elements. We chose Cartesian meshes for their simple structure in space distribution. However, since we use Cartesian grids, they do not necessarily fit with the considered internal discontinuities as long as a certain refinement is reached. Our contribution in this field is an adaptive mesh refinement (AMR) method. The adaptive grid's refinements propose to work on simple discrete spaces on the graphic plane. For this reason, they are represented by regular figures that are easy to manage. We conceived the numerical domain with squares and cubes in two and three dimensions, respectively; each square part was represented as a node of a quadtree, or octree. In this thesis we analysed the duality of the grid, what is visually represented, and the tree, which is a useful computational tool for our purpose. The AMR approach allowed us to obtain accuracy in some parts of the numerical domain, facing the problem of internal discontinuities for a variable coefficient Poisson equation with a zoom in on regions of interest. This is in general a common purpose of AMR approaches if compared to uniform refinements that can result in being extremely costly. The use of quadrees and octrees for dynamic refinement methods has become increasingly common, in particular for solving PDEs, as in our case. The advantages of these structures as much as the technical cares they need are presented. The spatial discretization in this work was optimized in parallel. The main communications parts were almost automatic for construction and balancing of the mesh; however, the ordering of these structures is not at all immediate: it overcomes proximity problem on one hand, but creates sparsity for the discretization on the other. This argument required an analysis; thus, we devoted a description of the possible orderings, giving comparisons and further details about the chosen one: the Morton code. We explained the consequences in code development, in particular the sparsity of the discretization matrix that corresponds to the numerical resolution of the Poisson problem. Moreover, we presented the manner to handle this difficulty in parallel by providing a description of the solver used. We tested the performed code in its parallel capability by giving a study of weak and strong scalability and by highlighting the advantages in time, memory and precision of our AMR approach; we could conclude, in general, that several proven advantages broadly overcome the disadvantages given by the discretization.

The method proposed in this work was a finite-difference discretization that locally treat the non conforming regions; each configuration of the neighbourhood for the concerned octant was approached by the resolution of a minimisation problem in order to optimize the truncation error. We were able to ensure first-order of convergence; moreover, as the local linear systems that we devised minimize the deviation from second-order accuracy, we could present the global tendency to second order of our method. We explored the existent finite-difference methods conceived similarly, or

on similar meshes, if compared with ours. The method proposed in the current work is among the first finite-difference schemes to devise weights stencils without the use of internal interpolations and fictitious points. We also verified that our method can be applied on non-graded grids without changing its reasoning, and that, where first-order approaches like penalization are combined with our resolution, the computation retains the expected convergence. Our tests involved two and three dimensions, and completed the framework to demonstrate the good arrangement of the finite-difference method on hierarchical meshes with non-conforming zones. Future perspectives in this study are to combine our method to geometries and discontinuities more complex than those presented, considering meshes that can variate in time, thus implementing a dynamic AMR approach.

Concentrated solar power has the desirable properties of being a clean, renewable, and a sustainable energy source. The ability to store heat and produce power beyond daylight hours is their unique advantage in contrast to other renewable energy sources. In order to provide electricity during off hours for the system, phase-changing materials (PCMs) have become popular. The purpose is not only to avoid the natural absence of solar resource during night (~ 15) hours, but also to produce power during this time. One objective of this thesis was to devote the method proposed to the application of heat diffusion in hybrid materials with phase-changing properties. Also if the latent heat of these materials makes them interesting to design, the process described as thermal energy storage (TES) requires a complete study to be optimal. Moreover, the existent tested TES systems are not cost competitive with sensible heat storage; different candidates for the host material, coupled with PCM encapsulations, have been studied, such as metallic meshes and carbon fibres. We offered, among our tests, results about the time of charge with respect to the surface occupied by both media: the PCM salt and the foam of graphite in our case; a future perspective is to extend this study in three dimensions. We simulated two-dimensional sections of heat diffusion in one of all possible cases; the future interest would be to compare other materials in order to optimize in two and three dimensions an entire TES system such that: the molten capsules are able to liquefy totally during day hours, but not too fast to avoid the possibility of evaporation.

This objective requires further analysis, not only on heat propagation due to the percentage of space occupied by the PCM compared to its container, but also on the manner in which the capsules interact among themselves following positions, dimensions, physical properties (such as conductivity, capacity and latent heat), and so on. The contribution given in the last chapter of this thesis constitutes a basis for more complex studies; the most improvement done for heat conduction applications on composite media is to create simulations with different geometries and parameters for the con-

cerned materials, which allows us to provide, in the future, more detailed simulations in order to optimize the problems that, at present, involve concentrated solar power in thermal energy storage systems.

Bibliography

[1] A. P. Selvadurai, *Partial Differential Equations in Mechanics 1:Fundamentals, Laplace's Equation, Diffusion Equation, Wave Equation*, vol. 1. Springer Science & Business Media, 2013.

[2] A. P. Selvadurai, *Partial Differential Equations in Mechanics 2: the Biharmonic Equation, Poisson's Equation*, vol. 2. Springer Science & Business Media, 2013.

[3] M. Stone and P. Goldbart, *Mathematics for physics: a guided tour for graduate students*. Cambridge University Press, 2009.

[4] G. Barton, *Elements of Green's functions and propagation: potentials, diffusion, and waves*. Oxford University Press, 1989.

[5] S. Agmon, *Lectures on elliptic boundary value problems*, vol. 369. American Mathematical Soc., 2010.

[6] O. Pantz and C. Acad, "Sensibilité de l'équation de la chaleur aux sauts de conductivité," *Comptes Rendus Mathematique*, vol. 341, no. 5, pp. 333–337, 2005.

[7] M. M. Brahim, *Méthodes d'éléments finis pour le problème de changement de phase en milieux composites*. PhD thesis, Université de Bordeaux, 2016.

[8] P. Grisvard, *Elliptic problems in nonsmooth domains*. SIAM, 2011.

[9] C. S. Peskin, "The immersed boundary method," *Acta numerica*, vol. 11, pp. 479–517, 2002.

[10] A. Gilmanov and F. Sotiropoulos, "A hybrid cartesian/immersed boundary method for simulating flows with 3d, geometrically complex, moving bodies," *Journal of Computational Physics*, vol. 207, no. 2, pp. 457–492, 2005.

[11] P. McCorquodale, P. Colella, D. P. Grote, and J.-L. Vay, "A node-centered local refinement algorithm for poisson's equation in complex

geometries," *Journal of Computational Physics*, vol. 201, no. 1, pp. 34–60, 2004.

[12] A. McKenney, L. Greengard, and A. Mayo, "A fast poisson solver for complex geometries," *Journal of Computational Physics*, vol. 118, no. 2, pp. 348 – 355, 1995.

[13] F. Gibou, R. P. Fedkiw, L.-T. Cheng, and M. Kang, "A second-order-accurate symmetric discretization of the poisson equation on irregular domains," *Journal of Computational Physics*, vol. 176, no. 1, pp. 205–227, 2002.

[14] S. Osher and J. A. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations," *Journal of computational physics*, vol. 79, no. 1, pp. 12–49, 1988.

[15] J. A. Sethian, *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, vol. 3. Cambridge university press, 1999.

[16] C. Min, F. Gibou, and H. D. Ceniceros, "A supra-convergent finite difference scheme for the variable coefficient poisson equation on non-graded grids," *Journal of Computational Physics*, vol. 218, no. 1, pp. 123–140, 2006.

[17] F. Gibou, C. Min, and R. Fedkiw, "High resolution sharp computational methods for elliptic and parabolic problems in complex geometries," *Journal of Scientific Computing*, vol. 54, no. 2, pp. 369–413, 2013.

[18] J.-S. Huh and J. A. Sethian, "Exact subgrid interface correction schemes for elliptic interface problems," *Proceedings of the National Academy of Sciences*, vol. 105, no. 29, pp. 9874–9879, 2008.

[19] D. P. Young, R. G. Melvin, M. B. Bieterman, F. T. Johnson, S. S. Samant, and J. E. Bussoletti, "A locally refined rectangular grid finite element method: Application to computational fluid dynamics and computational physics," *Journal of Computational Physics*, vol. 92, no. 1, pp. 1 – 66, 1991.

[20] H. Johansen and P. Colella, "A cartesian grid embedded boundary method for poisson's equation on irregular domains," *Journal of Computational Physics*, vol. 147, no. 1, pp. 60–85, 1998.

[21] A. N. Marques, J.-C. Nave, and R. R. Rosales, "High order solution of poisson problems with piecewise constant coefficients and interface jumps," *Journal of Computational Physics*, vol. 335, pp. 497–515, 2017.

[22] X. Li, J. Lowengrub, A. Rätz, and A. Voigt, "Solving pdes in complex geometries: a diffuse domain approach," *Communications in mathematical sciences*, vol. 7, no. 1, p. 81, 2009.

[23] A. Guittet, M. Lepilliez, S. Tanguy, and F. Gibou, "Solving elliptic problems with discontinuities on irregular domains–the voronoi interface method," *Journal of Computational Physics*, vol. 298, pp. 747–765, 2015.

[24] A. Guittet, M. Theillard, and F. Gibou, "A stable projection method for the incompressible navier–stokes equations on arbitrary geometries and adaptive quad/octrees," *Journal of Computational Physics*, vol. 292, pp. 215–238, 2015.

[25] A. J. Chorin, "A numerical method for solving incompressible viscous flow problems," *Journal of Computational Physics*, vol. 135, no. 2, pp. 118 – 125, 1997.

[26] F. Losasso, F. Gibou, and R. Fedkiw, "Simulating water and smoke with an octree data structure," in *ACM Transactions on Graphics (TOG)*, vol. 23, pp. 457–462, ACM, 2004.

[27] S. Popinet, "Gerris: a tree-based adaptive solver for the incompressible euler equations in complex geometries," *Journal of Computational Physics*, vol. 190, no. 2, pp. 572–600, 2003.

[28] Z. Li, C. Wang, *et al.*, "A fast finite differenc method for solving navier-stokes equations on irregular domains," *Communications in Mathematical Sciences*, vol. 1, no. 1, pp. 180–196, 2003.

[29] E. Fadlun, R. Verzicco, P. Orlandi, and J. Mohd-Yusof, "Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations," *Journal of computational physics*, vol. 161, no. 1, pp. 35–60, 2000.

[30] M. A. Olshanskii, K. M. Terekhov, and Y. V. Vassilevski, "An octree-based solver for the incompressible navier–stokes equations with enhanced stability and low dissipation," *Computers & Fluids*, vol. 84, pp. 231–246, 2013.

[31] A. McAdams, E. Sifakis, and J. Teran, "A parallel multigrid poisson solver for fluids simulation on large grids," in *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pp. 65–74, Eurographics Association, 2010.

[32] R. A. Finkel and J. L. Bentley, "Quad trees a data structure for retrieval on composite keys," *Acta Informatica*, vol. 4, no. 1, pp. 1–9, 1974.

[33] S. F. Frisken and R. N. Perry, "Simple and efficient traversal methods for quadtrees and octrees," *Journal of Graphics Tools*, vol. 7, no. 3, pp. 1–11, 2002.

[34] A. D'Angelo, "A brief introduction to quadtrees and their applications,"

[35] H. Samet, "Hierarchical spatial data structures," *Design and Implementation of Large Spatial Databases*, pp. 191–212, 1990.

[36] H. Samet, "An overview of quadtrees, octrees, and related hierarchical data structures," in *Theoretical Foundations of Computer Graphics and CAD*, pp. 51–68, Springer, 1988.

[37] M. R. Lattanzi, *Table-driven quadtree traversal algorithms*. PhD thesis, Virginia Tech, 1989.

[38] H. Samet, "The quadtree and related hierarchical data structures," *ACM Computing Surveys (CSUR)*, vol. 16, no. 2, pp. 187–260, 1984.

[39] E. Balaras and M. Vanella, "Adaptive mesh refinement strategies for immersed boundary methods," in *47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition*, p. 162, 2009.

[40] P. M. Campbell, K. D. Devine, J. E. Flaherty, L. G. Gervasio, and J. D. Teresco, "Dynamic octree load balancing using space-filling curves," *Williams College Department of Computer Science, Tech. Rep. CS-03-01*, 2003.

[41] T. Asano, D. Ranjan, T. Roos, E. Welzl, and P. Widmayer, "Space-filling curves and their use in the design of geometric data structures," *Theoretical Computer Science*, vol. 181, no. 1, pp. 3–15, 1997.

[42] G. M. Morton, *A computer oriented geodetic data base and a new technique in file sequencing*. International Business Machines Company New York, 1966.

[43] C. Burstedde, L. C. Wilcox, and O. Ghattas, "p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees," *SIAM Journal on Scientific Computing*, vol. 33, no. 3, pp. 1103–1133, 2011.

[44] D. Causon and C. Mingham, *Introductory finite difference methods for PDEs*. Bookboon, 2010.

[45] D. N. Arnold, "Lecture notes on numerical analysis of partial differential equations," 2012.

[46] R. J. LeVeque, *Finite difference methods for ordinary and partial differential equations: steady-state and time-dependent problems*. SIAM, 2007.

[47] R. E. White, *Computational Mathematics: Models, Methods, and Analysis with MATLAB® and MPI*, vol. 35. CRC Press, 2015.

[48] M. J. Berger and P. Colella, "Local adaptive mesh refinement for shock hydrodynamics," *Journal of computational Physics*, vol. 82, no. 1, pp. 64–84, 1989.

[49] M. J. Berger and J. Oliger, "Adaptive mesh refinement for hyperbolic partial differential equations," *Journal of computational Physics*, vol. 53, no. 3, pp. 484–512, 1984.

[50] O. V. Vasilyev, "High order finite difference schemes on non-uniform meshes with good conservation properties," *Journal of Computational Physics*, vol. 157, no. 2, pp. 746–761, 2000.

[51] F. Ham, F. Lien, and A. Strong, "A fully conservative second-order finite difference scheme for incompressible flow on nonuniform grids," *Journal of Computational Physics*, vol. 177, no. 1, pp. 117–133, 2002.

[52] K. W. Morton and D. F. Mayers, *Numerical solution of partial differential equations: an introduction*. Cambridge university press, 2005.

[53] D. N. Arnold, "Stability, consistency, and convergence of numerical discretizations," in *Encyclopedia of Applied and Computational Mathematics*, pp. 1358–1364, Springer, 2015.

[54] P. D. Lax and R. D. Richtmyer, "Survey of the stability of linear finite difference equations," *Communications on pure and applied mathematics*, vol. 9, no. 2, pp. 267–293, 1956.

[55] M. Kadri, "Analysis of the nine-point finite difference approximation for the heat conduction equation in a nuclear fuel element," 1983.

[56] Z. Chen, D. Cheng, W. Feng, and T. Wu, "An optimal 9-point finite difference scheme for the helmholtz equation with pml," vol. 10, no. 2, p. 389–410, 2013.

[57] C.-H. Jo, C. Shin, and J. H. Suh, "An optimal 9-point, finite-difference, frequency-space, 2-d scalar wave extrapolator," *Geophysics*, vol. 61, no. 2, pp. 529–537, 1996.

[58] C. Batty, "A cell-centred finite volume method for the poisson problem on non-graded quadtrees with second order accurate gradients," *Journal of Computational Physics*, vol. 331, pp. 49–72, 2017.

[59] M. Cisternino, A. Iollo, L. Weynans, A. Colin, and P. Poulin, "Electrostrictive materials: modelling and simulation ," in *7 th European Congress on Computational Methods in Applied Sciences and Engineering* , (Hersonissos, Greece), ECCOMAS, June 2016.

[60] P. Angot, C.-H. Bruneau, and P. Fabrie, "A penalization method to take into account obstacles in incompressible viscous flows," *Numerische Mathematik*, vol. 81, no. 4, pp. 497–520, 1999.

[61] G. Carbou and P. Fabrie, "Boundary layer for a penalization method for viscous incompressible flow," *Advances in Differential Equations*, vol. 8, no. 12, pp. 1453–1480, 2003.

[62] F. Chantalat, C.-H. Bruneau, C. Galusinski, and A. Iollo, "Level-set, penalization and cartesian meshes: A paradigm for inverse problems and optimal design," *Journal of Computational Physics*, vol. 228, no. 17, pp. 6291–6315, 2009.

[63] S. Balay, S. Abhyankar, M. Adams, J. Brown, P. Brune, K. Buschelman, V. Eijkhout, W. Gropp, D. Kaushik, M. Knepley, *et al.*, "Petsc users manual revision 3.5," tech. rep., Argonne National Laboratory (ANL), Argonne, IL (United States), 2014.

[64] R. W. Freund, G. H. Golub, and N. M. Nachtigal, "Iterative solution of linear systems," *Acta numerica*, vol. 1, pp. 57–100, 1992.

[65] K. Morikuni, L. Reichel, and K. Hayami, "Fgmres for linear discrete ill-posed problems," *Applied Numerical Mathematics*, vol. 75, pp. 175–187, 2014.

[66] Y. Saad, "A flexible inner-outer preconditioned gmres algorithm," *SIAM Journal on Scientific Computing*, vol. 14, no. 2, pp. 461–469, 1993.

[67] V. Voller, "An overview of numerical methods for solving phase change problems," *Advances in numerical heat transfer*, vol. 1, no. 9, pp. 341–380, 1997.

[68] F. Jelassi, M. Azaïez, and E. Palomo Del Barrio, "A substructuring method for phase change modelling in hybrid media," *Computers and Fluids*, vol. 88, no. Complete, pp. 81–92, 2013.

[69] X. Jin, H. Hu, X. Shi, X. Zhou, and X. Zhang, "Comparison of two numerical heat transfer models for phase change material board," *Applied Thermal Engineering*, 2017.

[70] V. Voller, "Fast implicit finite-difference method for the analysis of phase change problems," *Numerical Heat Transfer*, vol. 17, no. 2, pp. 155–169, 1990.

[71] V. Voller and C. Swaminathan, "Eral source-based method for solidification phase change," *Numerical Heat Transfer, Part B Fundamentals*, vol. 19, no. 2, pp. 175–189, 1991.

[72] T. Kim, D. M. France, W. Yu, W. Zhao, and D. Singh, "Heat transfer analysis of a latent heat thermal energy storage system using graphite foam for concentrated solar power," *Solar Energy*, vol. 103, pp. 438–447, 2014.

[73] J. Brusche, A. Segal, C. Vuik, and H. Urbach, "A comparison of enthalpy and temperature methods for melting problems on composite domains," in *Numerical Mathematics and Advanced Applications*, pp. 585–592, Springer, 2006.

[74] P. Giménez, A. Jové, C. Prieto, and S. Fereres, "Effect of an increased thermal contact resistance in a salt pcm-graphite foam composite tes system," *Renewable Energy*, vol. 106, pp. 321–334, 2017.

[75] Z. Acem, J. Lopez, and E. P. Del Barrio, "Kno 3/nano 3–graphite materials for thermal energy storage at high temperature: Part i.–elaboration methods and thermal properties," *Applied Thermal Engineering*, vol. 30, no. 13, pp. 1580–1585, 2010.

List of Figures

List of Tables

Solution of the variable coefficient Poisson equation on Cartesian hierarchical meshes in parallel: applications to phase changing materials

Abstract

We consider problems governed by a linear elliptic equation with varying coefficients across internal interfaces. The solution and its normal derivative can undergo significant variations through these internal boundaries. We present a compact finite-difference scheme on a tree-based adaptive grid that can be efficiently solved using a natively parallel data structure. The main idea is to optimize the truncation error of the discretization scheme as a function of the local grid configuration to achieve second order accuracy. Numerical illustrations relevant for actual applications are presented in two and three-dimensional configurations.

Key Words: AMR, octree, finite difference, PDEs discretization, heat equation, Poisson equation, internal discontinuities.

Solution du problème de Poisson avec coefficients variables sur maillages Cartésiens hiérarchiques en parallèle: applications aux matériaux avec changement de phase

Résumé

On s'interesse aux problèmes elliptiques avec coefficients variables à travers des interfaces intérieures. La solution et ses dérivées normales peuvent subir des variations significatives à travers les frontières intérieures. On présente une méthode compacte aux différences finies sur des maillages adaptés de type octree conçues pour une résolution en parallèle. L'idée principale est de minimiser l'erreur de troncature sur la discretisation locale, en fonction de la configuration du maillage, en rapprochant une convergence à l'ordre deux. On montrera des cas 2D et 3D des résultat liés à des applications concrètes.

Mots clés: AMR, octree, differences fines, discrétisation équations aux dérivées partielles, équation de la chaleur, discontinuités intérieures.